Counterfactual causal analysis on structured data

Swarna kamal paul¹, Tauseef Jamal Firdausi¹, Saikat Jana¹, Arunava Das¹ and Piyush Nandi¹

¹ Tata Consultancy Services, Gitanjali Park, Kolkata, India
{swarna.kpaul, tauseef.firdausi, saikat.jana,
arunava.das10, nandi.piyush}@tcs.com

Abstract. Data generated in a real-world business environment can be highly connected with intricate relationships among entities. Studying relationships and understanding their dynamics can provide deeper understanding of business events. However, finding important causal relations among entities is a daunting task with heavy dependency on data scientists. Also due to fundamental problem of causal inference it is impossible to directly observe causal effects. Thus, a method is proposed to explain predictive causal relations in an arbitrary linked dataset using counterfactual type causality. The proposed method can generate counterfactual examples with high fidelity in minimal time. It can explain causal relations among any chosen response variable and an arbitrary set of independent causal variables to provide explanations in natural language. The evidence of the explanations is shown in the form of a summarized connected data graph.

Keywords: Counterfactual causality, genetic algorithm, Explainable AI

1 Introduction

Data collected in a business enterprise contains all the story. Insights gained by obtaining causal answers for business events will help devising better future strategy. However, causality is a fleeting concept. Absolute causality in chaotic world is extremely difficult to find. Counterfactual type causality is at the highest level on the ladder of causation [1]. Constructing a black box model from the data can reveal causal relations by predicting consequences on simulated interventions and thereby finding the counterfactuals. A condensed set of causal explanations are derived from counterfactuals and provided in natural language for easy interpretation. The usability of the solution is three-fold. It reveals causal structure in the business process. It helps finding important features influencing a response variable and their range of influence. It provides evidence and details of explanations in form of a summarized data graph and enables deeper understanding of business dynamics.

Rest of the paper is organized as following. Section 2 provides a description of causal analysis method. Section 3 discloses experimental results followed by conclusion.

2 Counterfactual Causal Analysis

In the context of Artificial Intelligence (AI), Explainable AI (XAI) [2] can be defined as a method or techniques to explain the outcome of a black box ML model. In this context XAI has been used to find and explain causal influence of multiple independent variables on the response variable. The proposed method creates a best possible black box model in terms of accuracy of predicting the response variable with respect to the causal variables. Later the same model is used to find causal influence of each causal variables on the response variable by generating counterfactuals using perturbation method.

2.1 Generating counterfactual examples

Counterfactual examples are samples which are minimally modified with respect to the original sample to alter the predicted value by a model. Thus, counterfactual explanations provide statements as smallest changes required to alter certain predicted value or decision. Majority of current well-known XAI methods are based on feature attribution based [6, 12]. Wachter et. al. [7] proposed generating counterfactual examples can be represented as optimization problem by minimizing distance between original and counterfactual sample.

There are several limitations in the existing methods of generating counterfactuals [8] for serving the purpose in this context. Thus, an algorithm is proposed for generating counterfactuals which is model agnostic and based on gradient-free optimization named as *Genfact*. It can generate multiple counterfactuals at once and can-do amortized inference [8], thus making the process fast. Given a dataset it can find counterfactuals pairs closest to each other and the pairs may not exist in the original dataset. This feature is useful in this context as the given dataset used for generating counterfactuals may not contain enough samples around the classification boundary, but the proposed method can generate samples around the boundary.

Algorithm 1 states the *Genfact* algorithm for generating counterfactuals. The algorithm works for both categorical and numerical values. If the response variable is numeric it is divided into C classes by defining ranges for each class. The encoded feature data is clustered to K clusters. This is done to group the nearest neighbors which can be used as initial population for genetic algorithm. Each cluster are assigned a normalized diversity score which is proportional to the entropy of the predicted classes of each sample in a cluster. Higher diversity score signifies better mixture of samples from different classes. Genetic algorithm is run on each cluster in the order of normalized diversity score until 40% of the samples are covered. The crossover operation handles both categorical and numerical variables and adjusts them in a way to avoid creating non-feasible samples. The mutated numerical feature values are bounded by the range defined by maximum and minimum values in the sample set within the cluster. The categorical values are shuffled among available values in the samples within the cluster. This way it satisfies the actionability feature mentioned in [8]. The final output consists of counterfactual pairs of samples. *PermuteAttack* [9] also uses genetic algorithm to create counterfactuals however they cannot do amor-

tized inference and generate counterfactual sample only for the input sample. No separate way of handling categorical and numerical values is mentioned.

Algorithm 1

```

genfact(samples, model, C, maxoffspringsize, maxiterations, maxpopulationsize)
  response variable  $\leftarrow$  model.predict(samples)
  if response variable is numeric
    divide response variable in C classes such that each bucket range contains at least
     $N/|C|$  samples, where  $N \leftarrow$  Total #samples
    Create K clusters with feature samples, where  $K = \text{round}(\# \text{feature samples} /$ 
     $(\text{clustersize} * 2))$ 
    entropy of each cluster  $\leftarrow -\sum_{c \in C} p_c \log(p_c)$ ,  $p_c$  = probability of class  $c$  in the
    cluster
    cluster normalized diversity score  $\leftarrow$  entropy/ $\log(\text{size of cluster} + 1)$ 
    sort clusters in descending based on normalized diversity score
    initialize factials, counterfactuals
    processeddatasize = 0
    for each cluster k in K
      newfactials, newcounterfactuals  $\leftarrow$  genetic_algo(k, model, maxiterations,
      maxoffspringsize, maxpopulationsize)
      append newfactials to factials and newcounterfactuals to counterfactuals
      if processeddatasize  $\geq 0.4N$ 
        break
      processeddatasize  $\leftarrow$  processeddatasize + |k|
    return (factials, counterfactuals)

genetic_algo(facts, model, maxiterations, maxoffspringsize, maxpopulationsize)
  for i in 1 to maxiterations
    facts  $\leftarrow$  crossover(facts, model, maxoffspringsize)
    fitness, counterfacs  $\leftarrow$  calculate_fitness(facts)
    facts, counterfacs  $\leftarrow$  selectbest(fitness, facts, counterfacs, populationsize =
    min(maxpopulationsize, |facts|*2)
  return (facts, countefacts)

crossover(facts, model, maxoffspringsize)
  initialize offspringsize = 0
  while offspringsize < maxoffspringsize
    parent1  $\leftarrow$  randomly select one from facts
    feature  $\leftarrow$  randomly select a feature
    parent2  $\leftarrow$  randomly select one from facts having different predicted class from
    parent1
    if feature is categorical
      swap feature of parent1 with parent2
    else if feature is numeric
      feature of parent1  $\leftarrow$  (feature of parent1 + feature of parent2)/2

```

```

    offspring  $\leftarrow$  parent1
    class of offspring  $\leftarrow$  model.predict(offspring)
    if offspring present in facts
        continue
    add offspring in facts
    offspringsize  $\leftarrow$  offspringsize +1
    return facts

calculate_fitness(facts)
for each sample in facts
    fitness  $\leftarrow$  minimum Euclidean distance from other samples in facts having different predicted class
    counterfactuals  $\leftarrow$  other sample in facts having minimum distance from sample and different predicted class from that of sample
    return (fitness, counterfactuals)

select_best(fitness, facts, counterfactuals populationsize)
facts  $\leftarrow$  select top populationsize facts based on fitness
counterfactuals  $\leftarrow$  select top populationsize counterfactuals based on fitness
return (facts, counterfactuals)

```

2.2 Generating causal explanations

Causal explanations can be obtained by training a simplified surrogate decision tree on counterfactual examples. The generated tree is represented by internal nodes acting as decision points and each leaf node as final prediction of a unique class. Each path from root to leaf node outlines the features involved in that classification and the probability of the outcome occurring. All that information can now be formatted into facts/statements in natural language for ease of understanding.

The number of explanation statement generated is proportional to the number of leaf nodes present in the decision tree. It may become difficult to comprehend all the statements if the size of the explainer tree increases. So, reducing the number of statements and shortening the length of a single statement is necessary for summarizing the explanations so that causal insights can be quickly comprehended by a user. Different summarization approaches are proposed for each problem class.

In case of binary classification problem, there are only two classes. But same class appears multiple times in the statements. To summarize, for a specific class all conditions are grouped together from different statements. Within a group same feature may appear multiple times with different boundary condition. To shorten the length of a statement, conditions having common features are merged by superimposing the boundary conditions. Thus, multiple statements with repeating classes are converted to summarized statements consisting only two unique classes. In case of multiclass classification problems, statements for each unique class are sorted based on the probability score of the class which is derived from a specific statement. The statements with top 3 probability scores are finally selected as summarized statements. In regres-

sion problems, a statement does not give any probability score like classification. It gives some estimated value of the response variable. As all of these values are some estimations, so a confidence interval is calculated using the root mean squared error of each estimation. The statements with top 3 sample sizes are only considered.

Filtering out relevant statements are followed by shortening of statement length if required. A statement containing some categorical feature may have a long list of values which can lengthen the statement. So, the list of values is filtered based on their frequency of occurrence in the original data. Top 5 unique values with highest frequencies are considered and other values are omitted.

2.3 Generating evidence for explanations

To provide evidence for the explanations a data graph is generated by applying the filter conditions obtained from the explanations on top of the actual dataset and thereby summarizing it. The data evidence provides justification of the explanations and also at the same time allows users to get a deeper understanding of the entity relationship dynamics. The following algorithm is used to generate the evidence graph.

Algorithm 2

Filter dataset based on the top n conditions generated by the explainer tree

For each column C in dataset

if C is numeric

 divide values of C in k ranges such that each bucket contains at least N/k samples, where N is total number of samples

 add each range in the nodelist with sample size as node size

else if C is categorical

 select top k values of C based on sample size and add them into nodelist with sample size as node size

For each column C in dataset

if C is response variable

For each node of type C in nodelist

 find and add edges in edgelist with respect to all other node types in nodelist

 make edgeweight as number of observed samples for the relation

else if C is feature variable

For each node of type C in nodelist

 find and add edges in edgelist with respect to other node types in nodelist and satisfying entity relation with respect to C

 make edgeweight as number of observed samples for the relation

Normalize node size and edge weight

3 Experimental Results

Experiments have been done on a Facebook advertisement dataset [5]. Experiments are done to evaluate the counterfactual generation algorithm against prior arts. A case study also demonstrates how to find key performance indicators and causal relations with respect to value delivered by the advertisements. “Total_conversion” has been

chosen as the response variable. “clicks”, “spentperclick”, “age”, “gender”, “interest” and “impression” are chosen as feature variables. Categorical variables are encoded using m-estimator [3] encoding.

3.1 Evaluating counterfactual generation algorithm

The counterfactual generation algorithm is compared with two of the existing methods, namely DiCE [11] and ceml [10]. For generating counterfactuals in DiCE an artificial neural network model has been trained with the dataset which in turn serves as the base model. Counterfactuals are generated for randomly sampled original dataset. For ceml and Genfact, random forest classifier model has been used as base model. The response variable is converted to categorical variable using the method mentioned in algorithm 1. For ceml counterfactuals are generated for randomly sampled original dataset with randomly sampled different target class. Comparison has been done based on total runtime, average Euclidean distance between counterfactual pairs and entropy of predicted classes in all counterfactual pairs. The entropy measures the diversity of the counterfactual pairs.

Table 1. Comparison of counterfactual generation algorithm

Algorithm	Runtime in seconds	Average distance	Entropy
DiCE	7675.48	0.029	0.848
ceml	1297.47	255153.44	0.9729
Genfact	2.172	0.5030	0.8831

Overall, considering the runtime and other measures it can be concluded that Genfact exceeds in performance compared to prior arts.

3.2 Case study on generating causal explanations

The dataset has been run through the proposed method with XGBoost [4] serving as the black box model which in turn provided the causal explanations and the evidence data graph. From the top 3 generated causal statements as stated below, it is evident that Impressions are affecting the Total_Conversion most. Interest and Spentperclick are the 2nd most influencing factor. In general, higher the Impressions higher is the Total_Conversion.

Explanation 1: If *Impressions* > 920683.0 & *Spentperclick* > 1.483 and <= 1.673, then *Total_Conversion* will be in range 10.63 – 15.29, [sample size: 16.8%]

Explanation 2: If *Impressions* > 453229.125 and <= 920683.0 & *interest* are 16, 10, 29, 27, 15, then *Total_Conversion* will be in range 3.35 – 7.13, [sample size: 28.9%]

Explanation 3: If *Impressions* <= 453229.125 & *interest* are 16, 10, 29, 27, 15, then *Total_Conversion* will be in range 1.66 – 3.20, [sample size: 42.8%]

Fig. 1. Evidence data graph generated on Facebook advertisement data

Fig. 1 illustrates different sections of data graph generated as evidence. Fig 1a illustrates data graph with respect to nodes “Total_Conversion” and “Impression”. In general, higher “Total_Conversion” values have relation with higher “Impressions”. Fig. 1b shows relations with respect to “Total_Conversion” and “interest”. Lower “Total_Conversion” values mostly have strong relations with “interest 16”, “interest 15” and “interest 10”.

4 Conclusion

As per the claim the proposed method has been demonstrated to find and explain causal relations of a KPI with respect to arbitrary set to feature variables. The performance superiority of the counterfactual generation algorithm has also been established. The current work can be extended to encompass causal analysis of time series data generated from complex dynamical systems.

References

1. Pearl, Judea; Mackenzie, Dana (2018-05-15). The Book of Why: The New Science of Cause and Effect. Basic Books.
2. Goebel, R., Chander, A., Holzinger, K., Lecue, F., Akata, Z., Stumpf, S., ... & Holzinger, A. (2018, August). Explainable AI: the new 42?. In International cross-domain conference for machine learning and knowledge extraction (pp. 295-303). Springer, Cham.
3. Andersen, Robert (2008). Modern Methods for Robust Regression. Quantitative Applications in the Social Sciences. 152. Los Angeles, CA: Sage Publications.
4. Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
5. <https://www.kaggle.com/chrisbow/an-introduction-to-facebook-ad-analysis-using-r>
6. Mothilal, R. K., Mahajan, D., Tan, C., & Sharma, A. (2020). Towards Unifying Feature Attribution and Counterfactual Explanations: Different Means to the Same End. arXiv preprint arXiv:2011.04917.

7. Wachter, S., Mittelstadt, B., & Russell, C. (2017). Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.*, 31, 841.
8. Verma, S., Dickerson, J., & Hines, K. (2020). Counterfactual Explanations for Machine Learning: A Review. *arXiv preprint arXiv:2010.10596*.
9. Hashemi, M., & Fathi, A. (2020). PermuteAttack: Counterfactual Explanation of Machine Learning Credit Scorecards. *arXiv preprint arXiv:2008.10138*.
10. Artelt, A. (2019). CEML-Counterfactuals for Explaining Machine Learning models-A Python toolbox.
11. Mothilal, R. K., Sharma, A., & Tan, C. (2020, January). Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* (pp. 607-617).
12. <https://www.rulex.ai/rulex-explainable-ai-xai/>