Exploring Causal Inference Methods for Data Leakage Discovery

Sebastian Brarda
Center for Data Science
New York University
sb5518@nyu.edu

Maria Elena Villalobos Ponte
Center for Data Science
New York University
mvp291@nyu.edu

Abstract

We are interested in analyzing the problem of Information Data Leakage detection with Causal Learning algorithms. Data Leakage in Machine Learning is not an academically strictly defined concept but it refers to the presence of a predictive variable in the features of a ML model that will not be present at deployment time. For this variable to be leakage, it must temporally proceed the target variable and be a consequence of it. In this work, we theoretically define the data leakage problem as a Causal Inference problem in a structured way. After that, we propose an algorithm based on widely known causal discovery methods such as Additive Noise Models and Conditional Independence tests to determine whether Machine Intelligence can be used in this setting.

1 Introduction

Considerable progress has been achieved in recent years in developing methods for inferring causal relations on observational data, where experimentation and intervention is not possible. These methods work under different assumptions and data generation processes which make their applicability limited to certain contexts. In this work, we will review some of these methods and try to come up with an algorithm to detect a very common problem in the Data Science industry, which is Data Leakage.

Data Leakage is not a well academically defined concept, but was mentioned as “one of the top ten data mining mistakes” by (Nisbet, Elder, and Miner 2009). It refers to the existence of information in the data meant to train a Machine Learning model, which will not be present at the time of production and would cause the ML model to make unrealistically good predictions. Many authors like (Rosset, Perlich, and Liu 2007) have studied the phenomenon of leakage in several public ML competitions.

However, not many researchers have focused on defining the problem in a structured way, probably because its origin is usually a human mistake. (Kaufman et al. 2012) focused on defining the different types of data leakage, and coming with an analytical methodology to avoid it. However, the methodology is based on exploratory analysis and data management techniques. Data Leakage is a practical problem for which Machine Intelligence has not been used so far.

Supervised Machine Learning models exploit the dependence between variables to make predictions for the values of unknown variables given the values of known variables, but it does not make any assumptions about the causal relation between these variables. In other words, ML models applied in the industry usually do not make any assumptions about causality of the variables. Because of that, a Data Scientist that is not familiar with the data generating process of a particular dataset might only detect leakage by observing a ‘too-good-to-be-true’ performance of the model in a holdout set or by detecting high correlations in the exploratory analysis. But the distinction is ultimately made by human judgement most of the times.

If one of the supposedly explanatory features of the training dataset was a non-deterministic consequence of the target variable the leakage could potentially be harder to identify. Also, we would expect the model to have a significantly lower performance when deployed in a production environment, since the leaked information would not be present when the model is used to make real decisions. If a variable or information is consequence of the target variable, it must occur after the target variable is generated. That is why (Pyle 1999) decides to refer to leakage as ”Anachronisms”, meaning something that is out of place in time.

Because of this guiding principle, a method that helped identify the causal relation between the target variable and each of the remaining variables would allow us to identify leakage. The different possible causal relations generating dependence between two variables were first defined by (Reichenbach 1956), but can be summarized in the following base cases: i) X directly or indirectly causing Y ii) Y directly or indirectly causing X iii) X and Y causing each other mutually in a cyclic way iv) X and Y caused by T, a ”confounder” (Pearl 2009) v) X and Y are both causing Z

for which it is conditioned upon, ”selection bias”. The last case vi) would be the one where no significant dependence is observed between X and Y.

In particular, we are interested in determining the causal relation between each supposedly explanatory feature X_i and the target variable Y. If we can identify these relations, then we would be able to detect for example if Y is causing X_i (case ii) which would be a clear example of Data Leakage.

In the next section we will analyze in detail each of these possible pairwise causal relations between a feature X_i (which we will call X from now on) and a target variable Y and how do they relate with leakage. Later, we will describe some existing methods for causal discovery and propose a proof of concept application to detect leakage based on them. Finally, we will try this application on real and simulated data followed by a conclusion and discussion.

2 Problem Definition

In this section we will use the causal terminology introduced by (Pearl 2009), more specifically the concepts of intervention, the $do(X = x)$ operator, and Causal Graphs. We will analyze in detail the possible causal relations between two observed variables X, Y and a single latent variable as described in (Mooij et al. 2014) and how would they relate with leakage. Note that any pairwise causal inference analysis on the variables in the dataset of interest would correspond to one of these cases or a combination of them.

2.1 Feature directly or indirectly causes Target

$$\mathbb{P}_Y \neq \mathbb{P}_{Y|do(x)} = \mathbb{P}_{Y|x}$$

$$\mathbb{P}_X \neq \mathbb{P}_{X|do(y)} \neq \mathbb{P}_{X|y}$$

$$\begin{tikzcd} X \arrow[r] & Y \\ X \arrow[r] & Z \arrow[r] & Y \end{tikzcd}$$

In this case, we assume that there is no confounder (common parent), nor common child upon it was conditioned. If we intervened on X we would get a different distribution of resulting Y. This is true even if the causality is not direct (ie. there is one or more intermediate Z_i variables that are observed or unobserved, but not conditioned upon). For example if our target variable Y was current price of a real estate property and the explanatory variable X was the year of construction we would be clearly in this scenario, since fixing X would completely change the distribution of prices and clearly the year of construction occurred before the property being priced and then causes it. If there were intermediate variables Z, for example the fact that certain construction materials were not available before certain year which made constructions better after that year, would not affect the causal relation nor the intervention consequences. This is clearly a valid case for training a ML model that will be put in production, since no data leakage is present.

2.2 Target directly or indirectly causes Feature

$$\mathbb{P}_Y = \mathbb{P}_{Y|do(x)} \neq \mathbb{P}_{Y|x}$$

$$\mathbb{P}_X \neq \mathbb{P}_{X|do(y)} = \mathbb{P}_{X|y}$$

$$\begin{tikzcd} X & Y \arrow[l] \\ X & Z \arrow[l] & Y \arrow[l] \end{tikzcd}$$

This case is analogous to the previous one, but the Y target variable causes the X supposedly explanatory variable, then the X variable would not be present at the time the model was launched for production since it happens after Y. This is a clear case of data leakage. For example, if we were trying to predict which patients have high probability of having cancer in the future, we could try to build a model based on features that happened before certain patients having cancer, and because of that are potential causes of cancer. If in our dataset we had information about patients having oncologic prescriptions, we would be in presence of leakage, since clearly patients only take that kind of drugs as a consequence of having cancer.

2.3 Feature and Target are independent

$$\mathbb{P}_Y = \mathbb{P}_{Y|do(x)} = \mathbb{P}_{Y|x}$$

$$\mathbb{P}_X = \mathbb{P}_{X|do(y)} = \mathbb{P}_{X|y}$$

$$\begin{tikzcd} X & Y \end{tikzcd}$$

There exist also the case were the feature X and the target variable Y are independent. In this case, if we intervened on X the distribution of Y would not change, and the opposite is true as well. This is not a case of leakage even if X was generated before Y simply because even if at time of production X is not present, performance of the ML model should not be impacted negatively.

2.4 Feedback Loop

$$\mathbb{P}_Y \neq \mathbb{P}_{Y|do(x)} \neq \mathbb{P}_{Y|x}$$

$$\mathbb{P}_X \neq \mathbb{P}_{X|do(y)} \neq \mathbb{P}_{X|y}$$

$$\begin{tikzcd} X \arrow[r, bend left] & Y \arrow[l, bend left] \end{tikzcd}$$

This is the case when X causes Y and subsequently Y causes X. The example given in (Mooij et al. 2014) for this causal relation is that an increase in global temperature causes sea ice to melt, which then causes an increase in temperature. We somewhat disagree with this case, because these variables should be indexed by time, in which case the consequence is not the original variable but the one in the next time step. In any case we will assume that we observed variables correspond to a specific time index, there is no covariate shift and as a consequence there are no feedback loops.

2.5 Selection Bias

If feature X and target Y both have a common children for which we condition upon gathering the dataset, then we could fall in a common mistake of "detecting a spurious correlation". For example if the feature X is the number of times a person practices sports per week, and Y is the amount of calories consumed, the correlation between them would be very different if we sample from people that has a healthy weight/height ratio. This is because doing sports and eating healthy both cause a normal weight. Selection bias is often a statistical mistake in the way the data is gathered, or in the way the data is drawn. It is then, a human mistake most of the times, but a very different mistake from data leakage. Since in this work we will focus on data leakage, we will not consider this case and assume that there was no selection bias in the data. We will also make the assumption of iid. samples that all Machine Learning models do.

$$\mathbb{P}_{Y|s} \neq \mathbb{P}_{Y|do(x),s} = \mathbb{P}_{Y|x,s}$$
$$\mathbb{P}_{X|s} \neq \mathbb{P}_{X|do(y),s} = \mathbb{P}_{X|y,s}$$

$$\begin{tikzcd} X \arrow[dr] & & Y \arrow[dl] \\ & S & \end{tikzcd}$$

2.6 Observed or Latent Confounder

$$\mathbb{P}_{Y|s} = \mathbb{P}_{Y|do(x)} \neq \mathbb{P}_{Y|x,s}$$
$$\mathbb{P}_{X|s} \neq \mathbb{P}_{X|do(y)} \neq \mathbb{P}_{X|y,s}$$

$$\begin{tikzcd} & T \arrow[dl] \arrow[dr] & \\ X & & Y \end{tikzcd}$$

The final case to consider is when both the feature X and the target Y have a common parent. This parent can be observed or not, but it is not conditioned upon. In this case, the existence of the common parent might be the reason of correlation between X and Y but there is no causal relation between X and Y. For example feature X could be the number of e-commerce orders a person does per month, while Y is the amount spent on flight tickets over the summer. It is clear that both variables are caused by the income of the person. However, it is not clear that if we used the X to predict Y we would be in a leakage case. It is possible that we had the information of e-commerce orders of a given person for the months of January, February and March, and we are trying to predict the amount spent on flight tickets for the months of June, July and August. This would be valid, since the information of X was generated before Y, even if it is not the cause of Y. However, it is possible that X was generated at the same time of Y or even after, in which case the variable X would not be present at the time of production. Because of this, this case could be or could not be a case of data information leakage and we would not be able to tell only by identifying the causal relation. However, if a method exists that could identify this causal relation, that would assist the human decision when trying to understand if there was leakage or not, by restricting the number of pairwise relations to be analyzed to a smaller subset.

3 Methods

In this section we will review some causal discovery methods for observational data to come up with a proof of concept algorithm that would help us detect data leakage under certain assumptions. If the training dataset of interest consisted only of pairwise relation cases like 2.1 and 2.2, any pairwise causality classification algorithm such as Additive Noise Models or Information Geometric methods would be suitable. However, we cannot assume that no confounders will be present.

For example, if a certain feature X_i was both the cause of another feature X_j and target variable Y and there was no other causal relation between the variables, an ANM between X_j and Y could either conclude that X_j is the cause of Y or that Y is the cause of X_j and both conclusions would be wrong. Furthermore, in case there existed a confounding variable, it is possible that this variable was observed (present on the train dataset as variable X_i) or hidden (it is not present and we do not have evidence of its existence). For the purpose of this project, we will assume that all confounding variables are observed[1].

Also, we will assume that if there was an observed confounder in this form, the hypothesis space of the ML model would be big enough to capture the relation between parent variable X_i and target variable Y so that adding vari-

[1] We looked for different methods to account for hidden confounders but we concluded that these methods are still very unreliable and work only under very strong assumptions. We implemented a Python version of the **ICAN** algorithm (Janzing et al. 2009) which goal is to determine whether there is a hidden confounder by observing only a pair of variables. However this algorithm works only under very specific settings, so we decided to keep it out of our prototype.

able X_j into the model would not increase performance. In other words, we will only worry about the case where the pairwise causal classification algorithm would predict that target variable Y is causing variable X_j which would be a leakage false positive.

Another point to consider is whether the variables are continuous or random. Methods for causal discovery are different depending on the kind of variables that are being considered. For reasons explained in **Section 3.4.2** we will restrict our experiments to continuous variables.

3.1 Independence of Cause and Mechanism Assumption

Given random variables X and Y, their joint probability distribution admits two conditional decompositions:

$$\begin{aligned} p(x,y) &= p(x \mid y)p(x) \\ &= p(y \mid x)p(y) \end{aligned}$$

We can think about the conditional probabilities in the decomposition as mappings from the cause to the effect. Then, we need to decide between these two conditional distributions as the true causal mechanism. The Independence of Cause and Mechanism Assumption (ICM) suggests that we should prefer the causal direction under which the distribution of the cause is independent from the mechanism mapping the cause to the effect.

For example, if the ground truth is that $X \to Y$, then $p(x \mid y)$ and $p(x)$ must be independent and on the contrary $p(y \mid x)$ and $p(y)$ must not be independent. This asymmetry is exploited in most Observational Causal Inference methods, including the ones used in this paper.

3.1.1 Hilbert-Schmidt Independence Criterion (HSIC)

Introduced by (Gretton, Kenji Fukumizu, et al. 2007) the Hilbert-Schmidt Independence Criterion is a Kernel Statistical Test of Independence. The idea is to map each random variable to a reproducing kernel Hilbert space (RKHS) and look for correlations in that feature space. In particular, given a RKHS $\mathcal{F}$ from each $x \in X$ with corresponding kernel function $k(x, x') := \langle \phi(x), \phi(x') \rangle$ and another RKHS $\mathcal{G}$ from each $y \in Y$ with corresponding kernel function $l(y, y') := \langle \phi(y), \phi(y') \rangle$. The cross-covariance operator $C_{xy} : \mathcal{F} \to \mathcal{G}$ is defined such that for all $f \in \mathcal{F}$ and $g \in \mathcal{G}$

$$\langle f, C_{xy} \rangle = \mathbb{E}_{xy}([f(x) - \mathbb{E}_x(f(x))][g(y) - \mathbb{E}_y(g(y))])$$

The cross-covariance itself can be written as:

$$C_{xy} := \mathbb{E}_{xy}[(\phi(x) - \mu_x) \otimes (\phi(y) - \mu_y)]$$

Finally, if the kernel is characteristic (see K. Fukumizu et al. 2008) (Gaussian and Laplacian kernels are both characteristic) we have that:

$$\|C_{xy}\|^2_{HS} = 0 \iff X \perp Y$$

where the HS corresponds to the Hilbert-Schmidt norm.

3.2 First Approach - Additive Noise Models

This family of algorithms assumes certain structural equations of the following form:

$$S_i = f(Pa(x_i)) + n_i$$

Where $Pa(x_i)$ are the parents of x_i in a Causal Directed Acyclic Graph, meaning that are the causes of x_i and n_i are noise variables. Noise variables can also account for other exogenous hidden causes. The qualities of the functions $f_i(x)$ and the noise variables n_i are crucial for the identifiably of the model. For example, models are not identifiable if function $f_i(x)$ is linear with non-zero coefficient and the noises are Gaussian. But if the noises are non-Gaussian (Shimizu et al. 2006) or the functions are non-linear and smooth Mooij et al. 2014, then the models are identifiable if we assume minimality. For these models, causal minimality reduces to the condition that each function f_j is not constant in any of its arguments. Also, the probability density of the noise variables has to be strictly positive and the functions f_j have to be continuous.

What make ANM's reliable is the structure of the dependence between the causal variable and the noise. For example, if we define a variable $y = f(x) + n_i$ we can prove that if the noise is non-Gaussian or the function is non linear and $x \perp n_i$, we cannot create an inverted additive noise model of the form $x = h(y) + n'_i$ where $n'_i \perp y$. It is easy to see that in the pure Gaussian case we could invert the model, then identifiability is not possible.

In order to apply these models to real data there are two requirements: a consistent regression method and a consistent independence test. The idea is to train a regression to predict y with x and another one to predict x with y. Afterwards, we can score the level of independence of the residuals with the explanatory variables, and the one that has a higher measure of independence with the residuals is classified as the cause.

In (Mooij et al. 2014) several pairwise causality classification methods, including ANM, are benchmarked on real and synthetic data. Their empirical results show that these methods can sometimes be robust in the presence of hidden confounders, but they do not claim these methods are meant to be used in those cases. Nevertheless, one of the best performing methods they tried was the original ANM proposed by (Hoyer et al. 2009)

In terms of implementation, they used a Gaussian Process as the regressor (Rasmussen and Williams 2005) with a squared exponential covariance function, Gaussian noise likelihood and an Hilbert-Schmidt independence Criterion test. This method got an accuracy of 63 ± 10 and an AUC of 0.74 ± 0.05 on real and synthetic datasets. What we implemented is a slightly modified version of their algorithm (see Algorithm 1 of Mooij et al. 2014). The original version of the algorithm was meant to classify directionality between two variables assuming that one variable is causing the other one. Our version of the algorithm (please see **Appendix, Algorithm1**) classifies the directionality between each feature X_i and the target variable Y in three classes: i) $X_i \rightarrow Y$, ii)$X_i \leftarrow Y$, iii) $X_i \perp Y$ .Our algorithm not only classifies directionality of a pair of continuous random variables, but also conducts HSIC hypothesis testing between X and Y by approximating a Gamma distribution. In this way, we are able to detect if the variables are independent as well with certain confidence interval. We used $\alpha = 0.05$ for all the hypothesis tests. For the regression we used the pyGPs (Neumann et al. 2015) python implementation and implemented our own HSIC independence test on python.

3.3 Second Approach - Accounting for confounders

As mentioned before, an algorithm like ANM would suffice in case there were no confounders in the data. However, if one or more observed confounders X_i were the cause of both X_j and Y and there was no other causal relation between these variables , an additive noise model could incorrectly classify X_j as leakage. The problem could be fixed in case we where able to identify the confounding with a conditional independence test. In other words, if we find that X_j is a potential leakage in the dataset, we can afterwards compute a conditional independence test by conditioning on each possible subset of variables $X_{i \neq j}$ to account for possible confounders. If we can determine that $X_j \perp Y|S$ where S is some subset of the remainder variables, then we determine that it was a false positive. Otherwise, we determine that it was in fact leakage with a certain confidence.

This approach is similar in essence to the one taken by (Gretton, Peter Spirtes, and Tillman 2009) that introduces kPC a variant of the PC algorithm that assists structure learning with ANM. Both IC (Pearl 2009) and PC (P. Spirtes, Glymour, and Scheines 2000) algorithms rely on conditional independence tests and a set of deterministic rules to find the causal structure of the graph by assuming minimality and faithfulness. However, only some causal directions can be recovered completely with these types of algorithms, while some other dependencies remain directionally unspecified, or unconfirmed. The kPC algorithm uses kernel based hypothesis testing for conditional independencies and fits ANM to detect the causal direction once the dependencies are confirmed.

Since we are not interested in learning the whole graph structure, our algorithm (see **Appendix, Algorithm2**) only conducts conditional independence test between variable X_i and Y given all possible subsets S only if we suspect that there might be leakage[2] (if the ANM determines $X \leftarrow Y$.

3.4 Related work and other methods for causal discovery

Other causal discovery methods exist that could be applied for the purpose of detecting leakage in future work.

3.4.1 Information Geometric Causal Inference

If we assume a deterministic relation between a predictive variable X and an effect variable Y of the form $y = f(x)$ then ANM would fail in recovering the causal relation due to the lack of noise. These kind of methods on the opposite, rely on the form and invertibility of the $f(x)$ function. The Independence between Cause and Mechanism assumption is that the probabilistic density of the effect variable $p(y)$ should correlate with areas of small derivatives of the function $f'(x)$. Then if we look that the density of $p(x)$ gives no information about the inverse function $f'^{-1}(y)$ inferring that $x \rightarrow y$. These kind of models however do not perform well with real world noisy data.

3.4.2 Causality discovery for Discrete Random variables

Some experiments have been done for Discrete random variables. In particular, conditional independence tests such as Mutual information tests or Chi squared tests work particularly well for algorithms like PC and IC. Regarding ANM's, (Peters, Janzing, and Scholkopf 2011) proved that these methods can work with discrete random variables, but only under very strong assumptions and relatively high cardinality of the data. For Machine Learning settings like multi-class classification with multi-class features that are usually fed into the model as dummy variables, these methods are probably not very reliable. No ANM research has been done to try to find directionality between a discrete RV and a continous RV.

[2]In case the ANM did not determine that X_i is leaked, but the user still wanted to know if the causal relation with Y is only through confounders, this algorithm could be modified to do all conditional independence tests between each variable X_i and variable Y and then train an ANM in case X_i was not conditionally independent from Y. However, this would be computationally more expensive in most of the cases. This procedure should be conducted in case user believes that the hypothesis space of the ML model is not accurate for learning the relation between X_i and Y, but a third variable X_j that is a children from X_i is able to express that relation under that setting.

3.4.3 Machine Learning approaches

Other approaches like (Lopez-Paz, Muandet, and Recht 2014) proposed the Randomized Causation Coefficient which pose causality as a kernel mean embedding classification problem. The idea is to use ML learning to classify pairs of random variables and use the 6 possible relations between random variables described in section 2 as the labels. The probability distributions are featurized with kernel mean embeddings and random Fourier features. Lopez-Paz later suggested using Neural Networks for this classification task. Some recent approaches like (Chalupka, Eberhardt, and Perona 2016) have used Neural Networks to classify pairs of discrete random variables. The idea is that by training a supervised Neural Network model with pairs of variables where the causal direction is known, we can predict with good accuracy the causal direction on unseen samples. This approach looks promising but it has not been widely tested.

4 Experimental Evaluation

4.1 Cause and Effect Pairs

We conducted an experiment to evaluate the use of ANMs to classify the causal direction of pairs of variables. In this case we did not consider the possibility of independence between X and Y.

The first evaluation was on the 105 examples of real word data used by (Mooij et al. 2014) which is publicly available online. Out of these, only 93 were pairs and we got an accuracy of 66.6% which is consistent with performance reported by (Mooij et al. 2014) .

Later, we used the "Cause-Effect" pairs of the Kaggle Cause-Effect pairs competition 2013. It consists of 2000 pairs with the following labels: i) $X \rightarrow Y$, ii) $X \leftarrow Y$, iii) $X \perp Y$, iv) $X - Y$ meaning they have a confounder. We only considered 150 random pairs with labels i) or ii) and got an accuracy of 59%

We also tested the simulated data from (Peters, Janzing, and Scholkopf 2011, Appendix C) f the mentioned paper. The first set of pairs corresponds to samples from random densities using Gaussian Processes. In the second case, there is an unobserved confounder that causes both X and Y, but there is also a causal relation between them.

Table 1: Cause and Effect Results

Dataset	N_pairs	Accuracy
Real World Data pairs	93	0.66
Synthetic Kaggle	105	0.59
Synthetic Pairs	100	0.71
Synthetic Pairs Confounder	100	0.81

4.2 Synthetic data with leakage

In order to simulate a dataset in which one of the feature variables is *leaked*, we followed this data generating process:

$$X_1, \ldots, X_N \sim \mathcal{U}(-1, 1)$$
$$Y = f(X_1, \ldots, X_S) + n_Y$$
$$leak = g(Y) + n_L$$

We first establish a dataset size, we tested datasets with 200 and 600 data points. Then, we pick a number of variables N at random from 1 to 5. We sample their values from a uniform distribution between -1 and 1. Then, we pick a random transforming function for each variable from: identity, scale, exponential, square and cube. The target variable is then created by combining the N transformed selected features by either adding, subtracting, multiplying or dividing them (operations are as well selected randomly) and adding some noise n_Y. Finally, the leaked variable is built using a randomly selected transformation of the target variable, plus some noise n_L.

We tested two types of noise, Uniform and Gaussian as follows:

$$n_Y, n_L \sim \mathcal{U}(-0.01, 0.01)$$
$$n_Y, n_L \sim \mathcal{N}(0, 0.001)$$

After generating a dataset, the Algorithm 1 is used to classify each feature as leakage, independent or cause. We repeat the data generation process and ANM classification 100 times for each data size and type of noise. Please note that the number of features and the function that determines Y and $leak$ variables will vary each time. Then we compute the accuracy of our method to classify each type of variable using the number of correctly classified variables over the total number of variables of that class. The results are shown in Table 2 for each dataset size and type of noise.

Table 2: Synthetic Data Results

		Accuracy		
Num. points	**Noise**	**Leakage**	**Ind.**	**Cause**
200	Uniform	0.641	0.92	0.692
200	Gaussian	0.602	0.948	0.579
600	Uniform	0.680	0.90	0.566
600	Gaussian	0.622	0.926	0.602

The results are consistent with ANM performance shown in (Mooij et al. 2014). While the classification accuracy of Leakage seems to be slightly higher than the one of Causes, this results are not statistically significant. Also it is important to consider that the total number of causal variables evaluated is bigger than the number of leaked variables. Regarding the independent class, the accuracy is consistently high.

4.3 Evaluation of ANM in presence of Confounders

We conducted a small simulation with the following data generating process:

$$X_1, X_2 \sim \mathcal{U}(0, 10)$$

$$n_Y, n_L, n_C \sim \mathcal{U}(-1, 1)$$

$$Y = \log(X_1) + n_Y$$

$$X_3 = \log(X_1) + n_C$$

$$X_4 = Y^2 + n_L$$

So X_1 will be the parent and confounder of X_3 and Y, X_2 will be a variable independent from Y and X_4 would be a leaked variable. The only true cause of Y in this case is X_1

We draw 40 samples of 500 points each from this distribution and train **Algorithm 2** with each of these samples. All the settings of the algorithm were the same as in the previous experiment with the added component of the conditional independence test in case we detected leakage.

We repeated the experiment by changing the conditional independence test each time. We tried the following tests from the **bnlear** package from **R**: linear correlation (with and without Monte Carlo permutation test), Fisher's Z (with asymptotic normal test and MC permutation test, mutual information (with chi-square test and MC permutation test) and shrinkage estimator for the mutual information.

In terms of Leakage detection, the results were similar to the previous experiment with 70.7% accuracy in detection of the leaked variables. However, in the cases when the confounded variable X_3 was detected as leakage in the fourth step of the algorithm, none of the conditional independence tests were able to detect the real independence structure in a significant percentage of times, even with very small α values. Confounded variable X_3 was identified as leakage 58.5% of the times, but only a 4.2% of the times it was identified as conditionally independent of Y given X_1 with the best performing test (shrinkage estimator for the mutual information)

We think this is related to the fact that we are exploiting non-linearity of the functions and noises to train the models, and these conditional independence tests are clearly not powerful enough to detect these non-linear relations. One of the future improvements of this work would be to implement a more powerful non-linear test like the kernel conditional independence test[3] (cHSIC Sun et al. 2007)

[3]We actually are very close to finalizing an implementation of this test on Python. Unfortunately there is no implemented open source version on Python or R so far.

4.4 Evaluation on Real Data

Finding real datasets with known leakage is a difficult task. For example, it is well know that in **KDD-Cup 2008** which purpose was to detect cancer from mammography data, the "Patient ID" feature had a tremendous predictive power. Similarly, in the **INFORMS Data Mining challenge** held on the same year, participants were expected to predict pneumonia based on patient records from hospitals. In this case, "the target variable was embedded as a special value of one or more features in the data given to competitors" according to (Kaufman et al. 2012). But once the leakage problem is detected, the organizers of competitions quickly proceed to remove the leaked features, so finding these datasets was not possible.

4.4.1 Boston Dataset

The first experiment was to run **Algorithm1** on the very well know "Boston dataset" (Harrison and Rubinfeld 1978). This is a classic dataset use of regression in ML classes. It has 506 observations, 13 features, 12 of which are continuous and the target variable is the median value of owned occupied homes by suburb in the Boston area. We run the algorithm with the 12 continuous variables and non of the variables was detected as independent of the target variables with an $\alpha = 0.05$. Three of the variables were detected as leakage:

Table 3: Variables Identified as Leakage in Boston Dataset

As Leakage	**Diff. in Statistics**	**Description**
LSTAT	0.0011	% lower status of the population
RM	0.00065	average number of rooms per dwelling
NOX	8e-06	nitric oxides concentration (parts per 10 million)

The **NOX** variable is very difficult to interpret in its relation with the house pricing, and it was marked as leakage by a very small margin. But it is interesting to analyze the other two variables. A priori, one would think that the average price of the houses is encouraging poor people to live in cheap neighbourhoods and not the other way around (that a higher percentage of poor people in neighbourhoods determine the house prices), so it is possible that we are in fact in presence of leakage. Regarding the average number of rooms per house, the relation is not clear neither but the intuition is that a smaller number of rooms is causing prices to be lower. In this case we would be in presence of a false positive.

4.4.2 Generating Synthetic Leakage from Real Data

Our last simulation was to gather data from the Real World pairs mentioned in **section 4.1** and generate synthetic leakage from them. For this purpose we randomly selected 10 effect variables from the pairs. For each of these variables we created 100 leaked variables by randomly applying one of the transforming functions and adding one of the noises mentioned in **section 4.2**. The classification accuracy was 71.2%.

5 Conclusions and Discussion

We have proposed an additive noise model based approach to detect data information leakage in observational datasets meant to train a machine learning model. We found similar challenges to the ones that the causal learning inference field is used to: very limited real data to experiment which forced us to generate synthetic data. Furthermore, these experiments are computationally expensive (specially because of the Gaussian Process regression in the ANM) so our experiments were not very exhaustive nor statistically meaningful. Nevertheless, we obtained similar performance in our data leakage detection experiments to the ANM benchmarks in literature.

If we consider that current performance of these models is around $65 \pm 10\%$ we can quickly conclude that our system will not be extremely reliable in finding non-deterministic leaked variables. However, we have theoretically defined data leakage as a causal inference problem in a structured way, and designed an algorithm that detects it better than random.

We think that once the causal inference methods and conditional independence tests become more powerful they could be directly implemented into our algorithm an improve performance. In future work we would also try to explore other current promising methods, such as Neural Network[4] causal inference algorithms. Also, we think that probably the lowest performing component of the ANM implementation is the Gaussian Process regression, so we would like to try implementations with more powerful regressions in the future. Lastly, we would like to implement an enhanced version of **Algorithm 2** with a cHSIC conditional independence test to account for observed confounders.

It is also worth mentioning that we only explored the continuous case, meaning datasets destined to supervised regression tasks. However, many of the supervised ML tasks are classification or probability estimation, so it is worth exploring the discrete case in the future as well.

[4]Unfortunately these kind of pre-trained models are extremely data hungry and as we said, the amount of real world data for causal inference learning is very limited

References

blx@hook@bibinit

Reichenbach, Hans (1956). “The direction oftime”. In: *Los Angeles(California UP)*.

Harrison, David and Daniel L Rubinfeld (1978). “Hedonic housing prices and the demand for clean air”. In: *Journal of environmental economics and management* 5.1, pp. 81–102.

Pyle, Dorian (1999). *Data Preparation for Data Mining*. 1st. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN: 1558605290, 9781558605299.

Spirtes, P., C. Glymour, and R. Scheines (2000). *Causation, Prediction, and Search*. 2nd. MIT press.

Rasmussen, Carl Edward and Christopher K. I. Williams (2005). *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press. ISBN: 026218253X.

Shimizu, Shohei et al. (2006). “A Linear Non-Gaussian Acyclic Model for Causal Discovery”. In: *J. Mach. Learn. Res.* 7, pp. 2003–2030. ISSN: 1532-4435.

Gretton, Arthur, Kenji Fukumizu, et al. (2007). “A kernel statistical test of independence”. In: *Advances in neural information processing systems*, pp. 585–592.

Rosset, Saharon, Claudia Perlich, and Yan Liu (2007). “Making the Most of Your Data: KDD Cup 2007 ”How Many Ratings” Winner’s Report”. In: *SIGKDD Explor. Newsl.* 9.2, pp. 66–69. ISSN: 1931-0145. DOI: 10.1145/1345448.1345463.

Sun, X. et al. (2007). “A Kernel-Based Causal Learning Algorithm”. In: *Proceedings of the 24th International Conference on Machine Learning*. Max-Planck-Gesellschaft. New York, NY, USA: ACM Press, pp. 855–862.

Fukumizu, K. et al. (2008). “Kernel Measures of Conditional Dependence”. In: *Advances in neural information processing systems 20*. Max-Planck-Gesellschaft. Red Hook, NY, USA: Curran, pp. 489–496.

Gretton, Arthur, Peter Spirtes, and Robert E. Tillman (2009). “Nonlinear directed acyclic structure learning with weakly additive noise models”. In: *Advances in Neural Information Processing Systems 22*. Ed. by Y. Bengio et al. Curran Associates, Inc., pp. 1847–1855.

Hoyer, Patrik O. et al. (2009). “Nonlinear causal discovery with additive noise models”. In: *Advances in neural information processing systems*, pp. 689–696.

Janzing, Dominik et al. (2009). “Identifying confounders using additive noise models”. In: *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. AUAI Press, pp. 249–257.

Nisbet, Robert, John Elder, and Gary Miner (2009). *Handbook of Statistical Analysis and Data Mining Applications*. Academic Press. ISBN: 0123747651, 9780123747655.

Pearl, Judea (2009). *Causality*. en. Google-Books-ID: f4nuexsNVZIC. Cambridge University Press. ISBN: 978-0-521-89560-6.

Peters, Jonas, Dominik Janzing, and Bernhard Scholkopf (2011). “Causal inference on discrete data using additive noise models”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.12, pp. 2436–2450.

Kaufman, Shachar et al. (2012). “Leakage in Data Mining: Formulation, Detection, and Avoidance”. In: *ACM Trans. Knowl. Discov. Data* 6.4, 15:1–15:21. ISSN: 1556-4681. DOI: 10.1145/2382577.2382579.

Lopez-Paz, D., K. Muandet, and B. Recht (2014). “The Randomized Causation Coefficient”. In: *ArXiv e-prints*. arXiv: 1409.4366 [stat.ML].

Mooij, Joris M. et al. (2014). “Distinguishing cause from effect using observational data: methods and benchmarks”. In: *arXiv:1412.3773 [cs, stat]*. arXiv: 1412.3773.

Neumann, Marion et al. (2015). “pyGPs: A Python Library for Gaussian Process Regression and Classification”. In: *J. Mach. Learn. Res.* 16.1, pp. 2611–2616. ISSN: 1532-4435.

Appendices

A Proposed Algorithms for Leakage Detection

```
Data: X: features matrix, y: target vector
Result: DIR: Causality direction between each feature $X_i$ and $y$
Split data into training and test sets
$X\_train, Y\_train, X\_test, Y\_test \leftarrow SPLIT(X, y)$;
for $X_i \in X$ do
    if $\hat{C}(X_i, y) \implies X_i \perp Y$ then
        Establish direction;
        DIR[X$_i$] $\leftarrow (X_i \perp Y)$
    else
        1. Estimation:
        Fit Regression $x_{train_i} \mapsto \mathbb{E}(Y \mid X_{train_i} = x_{train_i})$;
        Get fitted model $\hat{f}_Y$;
        Fit Regression $y \mapsto \mathbb{E}(X_{train_i} \mid Y = y)$;
        Get fitted model $\hat{f}_X$;
        2. Residuals:
        $\hat{e}_y^{'} := y_{test} - \hat{f}_Y(X_{test_i})$;
        $\hat{e}_{x_i}^{'} := x_{test_i} - \hat{f}_X(Y)$;
        3. Calculate Independence Scores:
        $\hat{C}_{X_i \to Y} := \hat{C}(X_{test_i}, \hat{e}_y^{'})$;
        $\hat{C}_{Y \to X_i} := \hat{C}(y_{test}, \hat{e}_{x_i}^{'})$;
        4. Establish Direction:
        if $\hat{C}_{X_i \to Y} < \hat{C}_{Y \to X_i}$ then
            $DIR[X_i] := (X_i \to Y)$;
        else
            $DIR[X_i] := (Y \to X_i)$;
        end
    end
end
```

Algorithm 1: Pairwise Causal Direction Test

Where $\hat{C}(X, Y)$ is a consistent pairwise independence test

Where $\mathcal{P}(X_{-i})$ corresponds to the power set of all the other variables in the dataset.

```
**Data:** X: features matrix, y: target vector
**Result:** DIR: Causality direction between each feature $X_i$ and $y$
*Split data into training and test sets*
X_train, Y_train, X_test, Y_test $\leftarrow SPLIT(X, y)$
for $X_i \in X$ do
    if $\hat{C}(X_i, y) \implies X_i \perp Y$ then
        Establish direction;
    else
        1. Estimation
        2. Residuals
        3. Calculate Independence Scores
        4. Establish Direction
        5. Confounder Detection:
        if $DIR[X_i] == (Y \to X_i)$ then
            for $S \in \mathcal{P}(X_{-i}))$ do
                if $\hat{CI}(X_i, Y \mid S) \implies X_i \perp Y \mid S$ then
                    $DIR[X_i] := (Y \leftarrow S \to X_i)$;
                    break;
                end
            end
        end
    end
end
```

Algorithm 2: Pairwise Causal Direction Test with Observed Confounders