Time Robust Trees: Using Temporal Invariance to Improve Generalization

Luis Moneda[0000−0001−7153−4323] and Denis Mauá[0000−0003−2297−6349]

University of Sao Paulo, Brazil
`{luis.moneda, denis.maua}@usp.br`

Abstract. As time passes by, the performance of real-world predictive models degrades due to distributional shifts and learned spurious correlations. Typical countermeasures, such as retraining and online learning, can be costly and challenging in production, especially when accounting for business constraints and culture. Causality-based approaches aim to identify invariant mechanisms from data, thus leading to more robust predictors at the possible expense of decreasing short-term performance. However, most such approaches scale poorly to high dimensions or require extra knowledge such as data segmentation in representative environments. In this work, we develop the Time Robust Trees, a new algorithm for inducing decision trees with an inductive bias towards learning time-invariant rules. The algorithm's main innovation is to replace the usual information-gain split criterion (or similar) with a new criterion that examines the imbalance among classes induced by the split through time. Experiments with real data show that our approach improves long-term generalization, thus offering an exciting alternative for classification problems under distributional shift.

Keywords: Invariance · Generalization · Distributional Shift · Inductive bias

1 Introduction

Machine learning techniques are mainly evaluated by their ability to generalize, that is, to find valuable patterns from a training data sample that satisfactory apply to unseen instances [4]. Typically, that process involves a time dimension: training data refers to the past, while unseen instances come from the future. This temporal characteristic is usually dismissed by a time-stationary assumption of the data generating distribution. In practice, sampling distributions are seldom stationary, which causes spurious correlations to be learned and performance to degrade quickly.[1] A stereotypical anecdotal example is that of learning to classify an image of a husky dog as a wolf due to the presence of snow [1]. By blindly minimizing training error (or empirical risk), machine learning models

[1] There is often an inductive bias in learning algorithms towards estimating simpler accurate models. For complex tasks, it is often the case that spurious correlations are often simpler than non-spurious ones [32, 1].

absorb such relationships [1] and fail to generalize, even when a generalization promise from the validation stage is observed [8, 25].

A quick and dirty solution often employed is regularly retraining predictive models as new data arrives. However, this is unsatisfying from a business perspective, as labeling new data is often costly, and recurrently deploying new models into production can introduce inadvertent behavior and lead to significant harm. Also, business culture often is conservative towards adding or modifying existing systems, and such a constant update can decrease trust in machine learning models.

Spurious correlations can be defined as the non-causal statistical relationships between the target and non-target (covariate) variables [21, 20]. Thus, the impact of spurious correlations can be alleviated by incorporating causal reasoning into the learning process. However, performing causal inference in the absence of interventional data can be detrimental to predictive performance (hence to generalization) and is generally avoided unless the end task involves causal reasoning (such as producing counterfactuals). Recently, researchers have started advocating the benefits of ensuring some of the properties of causal inference for purely predictive problems without going through a full causal analysis [11].

One interesting property is invariance: causal relationships are invariant to change of environments, which are external settings of the covariates [23, 6]. By enforcing invariance in a learning algorithm, we regularize against spurious correlations and decrease the generalization error [1]. While samples from multiple environments are available in specific circumstances (e.g., clinical data collected at different health care centers), this type of data is missing and difficult to generate for most prediction tasks in the real world. Instead, a different type of information about the environment is present in the form of the temporal order in which observations are collected, often spanning a significant time period.

To circumvent the shortcomings highlighted and make use of the often abundant temporal information available in real industry datasets, in this work, we develop the Time Robust Trees (TRT), a new decision tree-inducing algorithm with a strong learning bias towards time-invariant predictive models. A TRT is obtained by modifying the standard recursive partitioning algorithm used to induce decision trees, replacing typical split criteria such as information gain or standard deviation with a new criterion that measures impurity across different time periods. We thus assume that the data is temporally ordered and the training set is segmented by, e.g., yearly data. A hyper-parameter defines the minimum number of examples by segment the model should keep as it learns new rules. This ensures that predictions on unseen data (which do not need temporal information) are more robust to spurious correlations in training data without requiring specific information about environments, causal relationships, or retraining.

Our experiments with seven real-world datasets show that when domain shift is significant, as measured by a domain classifier [24], there is a benefit to using TRTs as a base estimator for an ensemble instead of Decision Trees. The higher the change between the training period and future data, the higher the benefit of signaling to the model via the TRT design that we prefer to learn stable relationships.

The rest of the paper is organized as follows. We start in Section 2 reviewing related work in invariance learning, robust learning, and causal analysis. We then present our proposal in Section 3. We explain the experimental setup and present the empirical results with real data in Section 4. We conclude the paper in Section 5 with a discussion about the limitations of the proposed method and some possible improvements for the future.

2 Related Work

As discussed in the introduction, one way of hedging against spurious correlations is to explicitly consider causal relationships in model building. While there are many ways that can be achieved in the literature, a common approach is to resort to the principle of Independent Causal Mechanisms (ICM), which states that the causal generative process of a phenomenon is composed of autonomous modules that do not inform or influence each other [26]. In the probabilistic case, this means that the conditional distribution of each variable given its direct causes (i.e., its causal mechanism) does not inform or influence the other conditional distributions (mechanisms). Recall that an environment is defined as an external setting of the covariates and target variables. As such, data from different environments can be used to identify and learn the causal mechanisms and avoid learning spurious correlations. The hypothesis is that such causal mechanisms are time-invariant, hence improving the generalization ability of the model.

The Invariant Causal Prediction (ICP) [22] is a feature selection algorithm that finds the subset of causal features by testing if the error in the residual on this subset follows a property only found on the target variable's parents under the needed assumptions. ICP requires some mild conditions to be met and scales poorly to high-dimensional data.

The Invariant Risk Minimization (IRM) [1] exploits invariance without explicitly modeling causal relationships. Instead, it modifies the objective function to iterate in training environments and penalizes the lack of invariance across environments. A penalization term is derived for the case of linear classifiers (possibly after the input has been modified by a feature extractor), and the more general case of nonlinear (e.g., neural net) classifiers is left open. ICP requires data to be collected from different environments and annotated accordingly.

There are many other approaches designed to take advantage of the ICM principle. In the Recurrent Independent Mechanisms (RIM) network [10], attention [30] is used to activate different modules composed by RIMs. These modules learn different aspects of the problem, and it is expected that the invariant aspects will be useful when it needs to predict data that differ from the training distribution. Neural Causal Models [15] leverage known or unknown interventions in the observational data to learn. The weakly supervised disentanglement approach [16] learns how to disentangle components from high-dimensional feature spaces, like images, using the hypothesis that they are composed of a small number of relevant factors that can change, thus exploring the modularity of ICMs. The do-calculus in the presence of interventional data [3] is also used to tackle the problem of learning from available data from a few environments and generalizing to unseen environments.

In Robust Supervised-Learning (RSL) [2], the concept of an environment is explicitly missing, but the learning algorithm assigns weights to the training data and optimizes a worst-case scenario for such weights, hoping that such a scenario will also protect the performance at future unseen examples [12]. The first difference with respect to the model we propose here is that RSL considers an adversarial optimization problem while our approach considers worst-case optimization by segmenting the training data.

Our proposed learning algorithm is heavily inspired by the Causal Forest algorithm [31], which induces a decision tree from interventional data segmented into treatment and control groups. The algorithm enforces invariance by requiring a minimum number of examples from treatment and control groups at each split to contrast them in the leaves and reveal heterogeneous causal effects. In contrast, our proposed method assumes purely observational data and regularizes against spurious correlations and distributional shifts by requiring a minimum number of examples from every time period in every node of the decision tree.

There are also approaches that exploit temporal information as an environment proxy like ours in order to mitigate the effects of distributional shifts. The Temporal Decision Trees [14, 13] use timestamped data to induce a decision tree, targeting the construction of sequential predictions; as is the case with sequential prediction, the algorithm assumes time-dependence among examples with a stationary generating process. Our method instead makes the common assumption of independent and identically distributed data points.

3 Learning Time Robust Trees

Before formally describing the proposed algorithm, we will first motivate the necessity of time-robust learning methods and explain the limitations of current approaches with a toy example.

3.1 Motivational Example

Consider a setting with two finite-valued input variables X_1 and X_2, a binary target variable Y, and a time period variable T_{period}, used to segment the data into different diverse environments. We use three time periods to illustrate, thus $T_{period} = \{1, 2, 3\}$. Suppose we collect the data shown in Figure 1, where the data segments for $t = 1, 2$ consist of the available training set, and the data segment for $t = 3$ is observed after model deployment. We will call it the holdout set (note: this is not the typical validation dataset since we assume it is taken from a different distribution, arising possibly from a different environment and certainly from a different time period in the future). According to the example,

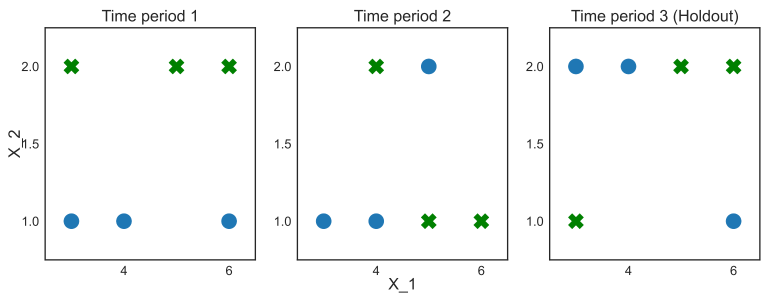

Fig. 1: An artificial example of data with spurious correlations. In the time period 1, there is an pure split on X_2 that leads to maximum accuracy in that data, but does much poorer performance for the time periods 2 and 3. The Time Robust Tree prefers the impure split on X_1, which provides more stable performance across different time periods.

X_1 is mildly predictive and stable for Y, while X_2 is a perfect predictor at $t = 1$ but irrelevant for $t = 2$ and $t = 3$. Thus, X_2 can be considered a spurious correlation or a non-static causal relation that shifted.

If the modeler uses all the available training data, a typical Decision Tree (DT) inducing algorithm will combine the data from periods 1 and 2 into a single training data set to evaluate the possible splits. In contrast, in the Time Robust Tree (TRT), as long as the modeler sets the period information as the environment, we consider the split performance separately when looking at every period. To illustrate it, we prune the example tree to have a single split in both cases. We use the Gini impurity (GI) minimization process in Table 1.

Table 1: Split evaluation process for the Decision Tree and the Time Robust Tree for the motivating example

		DT	TRT		
Variable	Split value	GI	GI at t=1	GI at t=2	Max. GI value
X_1	3	0.49	0.50	0.40	0.50
X_1	4	0.44	0.44	0.44	**0.44**
X_1	5	0.49	0.50	0.40	0.50
X_2	1	**0.27**	0.00	0.50	0.50

We use the Area Under the Curve (AUC) to evaluate the prediction quality. The measure goes from 0 to 1, and the higher, the better. By learning these splits, the Decision Tree achieves a 0.83 AUC on training but a poor result on holdout data of 0.50 AUC. The Time Robust Tree performs significantly worse in training, achieving an AUC of 0.67; however, it maintains that same performance in the holdout dataset, largely outperforming the Decision Tree. As this example shows, our proposal sacrifices training accuracy in the hopes of achieving superior performance on unseen data that suffer distributional shifts due to the presence of spurious correlations.

3.2 Time Robust Forests

We can now formally describe the Time Robust Tree induction algorithm. We denote an arbitrary impurity function used to evaluate the quality of a dataset split as L. The algorithm work as follows. Consider a timestamp column T_{stamp} representing the data point's capture time with the exact dimension of the random variables vectors $(X_1, ..., X_d, Y)$, where the X variables represent inputs and Y the variable of interest, that is, the target. The time period T_{period} is an aggregation of sequential examples when ordered by T_{stamp} using a human-centered concept, like hourly, daily, weekly, monthly, yearly, or simply putting together a fixed number of examples and reducing T_{stamp} granularity.

Given n time periods $T_{period} = t_1, t_2, \ldots, t_n$ in the training set, we find the best split s^* to divide the examples in X_{node} using the rule $X_f \leq v_f$ where f is a feature from all available features F at a certain value v_f from all possible values for the feature f in the training set V_f by applying recursively to every node data X_{node} until the constraints are not satisfied, being the first node the root containing all the training set:

$$\begin{aligned} s^* = & \min_{\forall f \in F, \forall v \in V_f} \max_{t \in T_{period}} L(X_{node}), \\ & \text{subject to } |X_{right,t}| \geq \rho \text{ and } |X_{left,t}| \geq \rho, \forall t \in T_{period} \,. \end{aligned} \tag{1}$$

The ρ is a scalar representing the minimum number of examples in every time period to perform a split. The model also accepts the average loss criteria.

$$\begin{aligned} s^* = & \min_{\forall f \in F, \forall v \in V_f} \frac{1}{|T_{period}|} \sum_{t=1}^{T_{period}} L(X_t), \\ & \text{subject to } |X_{right,t}| > \rho \text{ and } |X_{left,t}| > \rho, \forall t \in T_{period} \,. \end{aligned} \tag{2}$$

For the predictions $\hat{Y}$, the average from the leaf is taken without any consideration about the time period it belongs, $\hat{Y} = \frac{1}{|Y|} \sum y_i$.

It is worth isolating in the Equation 3 one of the differences from TRT. This period-wise score considers how the model performs in the different periods defined by the user to decide the optimal split. The other difference is the hyperparameter ρ. It interacts a lot with this part of the process—higher ρ guarantees a higher sample in each period for their evaluation regarding the split.

$$\frac{1}{|T_{period}|} \sum_{t=1}^{T_{period}} L(X_t) \,. \tag{3}$$

There is nothing particularly different in the step from Time Robust Tree to Time Robust Forest (TRF) in comparison to the one from a Decision Tree to a Random Forest [5]. Considering M trees, the final prediction $\hat{Y}$ becomes $\frac{1}{M}\sum_{m=1}^{M}\hat{Y}_m$, a random proportion of the input features F is considered when finding the best split for a node on Equation 1, and bootstrapping is performed in the training data before learning every tree.

3.3 Synthetic example

In order to see how TRT prevents spurious correlations from a causal perspective, consider the following artificial example. Once again, we include a spurious feature X_2 in the data generating process that makes the prediction non-stable in the training data. The example is extreme, since X_2 mimics Y in $t = 1$, while it is random in $t = 2$, both of them available for training. The X_2 keeps random in the following periods, consisting of the holdout set. It emulates the hypothesis that unstable properties are less likely to persist.

$$\begin{aligned} X_1 &\sim \mathbb{N}(0,1) \\ Y &\sim X_1 + \mathbb{N}(0,1) \\ X_2 &\sim f(e) \end{aligned} \tag{4}$$

where e is the time period variable, which is our environment. In the training, we have two training environments $\epsilon_{train} = \{1, 2\}$. The $f(e)$ defines X_2 following:

$$f(e) = \begin{cases} Y, \text{ if } e = 1 \\ \mathbb{N}(0,1), \text{ if } e \neq 1 \end{cases} \tag{5}$$

We make it a binary classification task by converting y to a positive class when greater than 0.5 and to the negative one otherwise. The holdout is composed of the following periods, starting at $t = 3$.

At first, we apply the TRT and the DT using similar hyper-parameters: 30 as maximum depth, 0.01 as minimum impurity decrease, 10 as a minimum sample by period for the TRT, and 20 as a minimum sample to split for the DT since we have two periods. The TRT presents an AUC of 0.83 in train and 0.81 in the holdout, while the DT performs around 0.92 AUC in training and 0.64 in the holdout. It shows how the TRT avoids learning from the spurious variable X_2, which lowers its training performance but makes it succeed in the holdout, while the DT goes in the opposite direction. However, we need to define the hyper-parameters following a process and objective criteria in a real-world case. In the following subsection, we show how to execute this step when using the TRT.

3.4 Hyper-parameter Optimization

When selecting hyper-parameters, a common strategy is to use the K-fold validation design [29]. However, during the hyper-parameter selection, this design pools

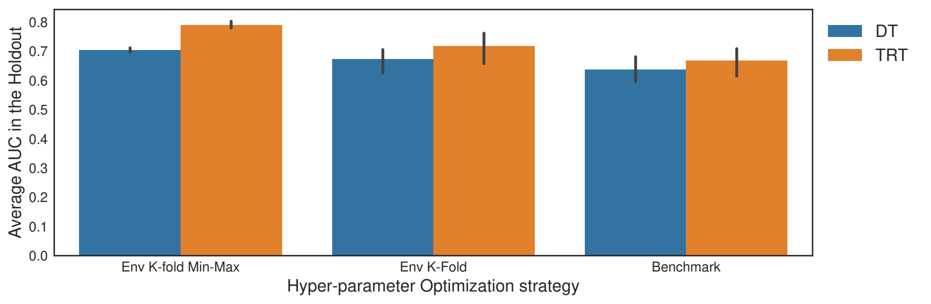

Fig. 2: A hyper-parameter optimization design that keeps the period-wise evaluation from the TRT algorithm is important to make the model keep its purpose of learning stable relationships.

the data from the periods and then select the set of parameters in which the performance is the highest. This process does not favor the period-wise design from TRT. We use a K-fold that generates folds containing just one environment, used as test folds to overcome it. We identify this approach as Environment K-Folds (Env K-Folds). Similar to what we use to learn the best split in the TRT. Besides taking the average performance in the folds to decide the hyper-parameters, we evaluate a second strategy when using the Env K-Folds. First, we average the performance in all folds consisting of the same environment and hyper-parameter set, then we group by only hyper-parameters sets and select the minimum performance, which is the worst environment case. Finally, we take the set with the highest performance among the worst cases to determine the model using the best worst case. We identify this approach as Env K-folds Min-Max.

We bootstrap the data and repeat the process ten times to evaluate these different designs. The results are the average of these ten best models following each approach. As seen in Figure 2, the TRT performs significantly better than the DT in the holdout set when using the Env K-folds Min-Max, while in the other two strategies, they are very similar.

4 Experiments

To validate the approach, seven public datasets in which a timestamp information and a reasonable time range are available were selected [18] [17] [19] [9] [27] [28] [7].

We split every dataset into two time periods: training and holdout. Then training period data is split randomly between training and test. For both benchmark and challenger, we use the Time Robust Forest python package.[2] The benchmark has all training examples with the same T_{period}, which is a special

[2] The source code and datasets used and install instructions are available on GitHub at (omitted due to blind reviewing).

case the TRF becomes a regular Random Forest. The challenger uses yearly or year-monthly segments.

In Table 2, it is possible to verify that the cases where TRF is an exciting challenger are the ones in which the benchmark has problems performing in the holdout as well as it does in the test. We train a domain classifier using the holdout as the target to clarify the evidence under scenarios the future data changes the most. The higher the AUC, the more significant the difference between test and holdout in that dataset. As seen in Figure 3, the results show the TRF performed better in the datasets with a more remarkable shift between training data and holdout data.

Table 2: Performance results. When comparing the AUC in the holdout from the TRF to the RF, the benchmark gets better performance on three cases. However, the difference between challenger and benchmark in the holdout always drops compared to the same difference in the test.

Dataset	Data split	Volume	Time range	RF	TRF	Δ TRF-RF
Kickstarter	Train	98k	2010-2013	**.736**	.717	-.019
	Test	24k	2010-2013	**.705**	.701	-.004
	Holdout	254k	2014-2017	.647	**.661**	.014
GE News	Train	21k	2015-2018	**.927**	.865	-.062
	Test	5k	2015-2018	**.879**	.839	-.040
	Holdout	58k	2019-2021	.805	**.821**	.017
20 News	Train	8k	-	**.939**	.869	-.070
	Test	2k	-	**.867**	.828	-.039
	Holdout	8k	-	.768	**.774**	.006
Animal Shelter	Train	75k	2014-2017	**.814**	.803	-.011
	Test	19k	2014-2017	**.792**	.790	-.002
	Holdout	61k	2018-2021	**.791**	.791	.000
Olist	Train	41k	2017	**.799**	.695	-.104
	Test	10k	2017	**.664**	.641	-.023
	Holdout	62k	2018	**.635**	.635	.000
Chicago Crime	Train	100k	2001-2010	**.936**	.909	-.027
	Test	61k	2001-2010	**.904**	.899	-.005
	Holdout	90k	2011-2017	**.905**	.902	-.003
Building Permits	Train	90k	2013-2015	**.990**	.984	-.006
	Test	22k	2013-2015	**.974**	.972	-.002
	Holdout	193k	2016-2017	**.977**	.973	-.004

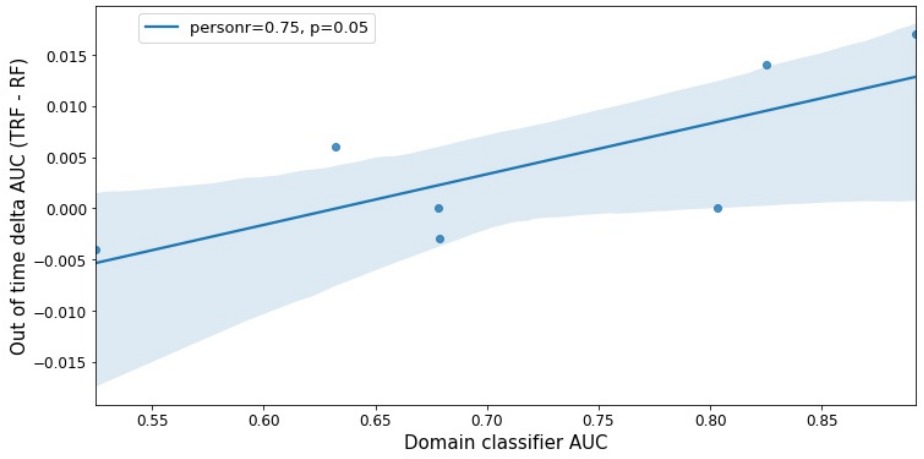

Fig. 3: Domain classifier performance by the delta improvement in the TRF. The greater the difference between the source and target data, translated by a high AUC for the domain classifier, the greater the benefit of learning invariant relationships to generalize to future unseen data.

5 Discussion and Conclusion

Ultimately, machine learning models are evaluated by their ability to generalize observed patterns to unseen data. In realistic scenarios, this often involves using the model under different conditions than those observed in the training stage, causing models learned by standard empirical risk minimization to perform unsatisfactorily when deployed. Common solutions such as constantly retraining are costly, unwanted from a business perspective, and may introduce inadvertent behavior in the system.

Typical real-world datasets are often collected during a significant period and contain temporal order information (i.e., timestamps) that is most often ignored during model construction. In this work, we proposed the Time Robust Trees, a new decision tree induction algorithm that uses temporal order to improve the generalization ability of predictions on unseen, future data. Our method segments data according to time and minimizes the variance of predictions across different time segments, delivering more time-stable models. Experiments with real-world data showing varying degrees of distributional shift suggest that Time Robust Forests are a promising alternative for applications where it is not possible to update the model continuously. Beyond the immediate practical purpose, the experiments show that exploiting time invariance as an inductive learning bias is attractive for non-sequential predictive tasks.

A main limitation of the proposed method is the requirements of temporal order (that is, a timestamp column in the dataset), a reasonable time range, and overlapping empirical distribution support for every period regarding the input features. Timestamp information is most commonly available in real-world datasets since every data is generated in time, and standard practice stores such information. Data collected during long time periods is not uncommon since (unlabeled), as businesses often store data continuously for significant amounts of time. Therefore, the most severe limitation is the need for overlapping empirical distribution support of the input features in every period. We can mitigate such a shortcoming by considering different time scales for the time periods while evaluating model performance and overlap. For example, consider an application that predicts customer acquisition success rate and has customer age as one of the input features. If the time period scale is too small, such as in days, it is very likely that certain age ranges be present in one time period and not in others. In the limit, each period would consist of a single example, causing the learned model to ignore age as a relevant predictive feature. By considering increasingly larger periods (say, of months or years), we can ensure that every data segment contains enough examples for every age range.

The proposed method identifies time periods and environments which are prone to problems. For example, considering two time periods as different environments while they were generated under the same environment will not degrade performance if data is sufficiently abundant but might do so if data is less abundant since it will require more data to meet the cut-off level in the splits. Instead, suppose we place two different environments in the same time period segment. In that case, we are losing an opportunity to offer the model two cases we want to keep relationships invariant and potentially enable the algorithm to create splits that are good only for one of the environments in the same period. However, we still want invariance between this period with the two environments and other periods in the training data. While fewer segments provides a higher volume of data in every period and enables learning more complex rules, it will also make it more likely that two different environments share the same period, compromising the inductive bias for invariant rules.

Real-world data with a good time range should offer enough flexibility to enable a period segmentation to overcome the requirement of overlapping input distribution support. For the future, we plan on exploring different automatic segmentation strategies, representation learning schema that satisfies the overlapping support requirement, combining boosting while respecting the invariance preference, and ensembles of differently regularized Time Robust Forests.

References

1. Arjovsky, M., Bottou, L., Gulrajani, I., Lopez-Paz, D.: Invariant risk minimization (2019)
2. Bagnell, J.A.: Robust supervised learning. In: AAAI. pp. 714–719 (2005)
3. Bareinboim, E., Pearl, J.: Transportability from multiple environments with limited experiments: Completeness results. Advances in neural information processing systems **27**, 280–288 (2014)
4. Bishop, C.M.: Pattern recognition and machine learning. springer (2006)
5. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)

6. Cartwright, N.: Two theorems on invariance and causality. Philosophy of Science **70**(1), 203–224 (2003)
7. of Chicago, C.: Chicago crime - bigquery dataset (2021), version 1. Retrieved March 13, 2021 from https://www.kaggle.com/chicago/chicago-crime
8. D'Amour, A., Heller, K., Moldovan, D., Adlam, B., Alipanahi, B., Beutel, A., Chen, C., Deaton, J., Eisenstein, J., Hoffman, M.D., Hormozdiari, F., Houlsby, N., Hou, S., Jerfel, G., Karthikesalingam, A., Lucic, M., Ma, Y., McLean, C., Mincu, D., Mitani, A., Montanari, A., Nado, Z., Natarajan, V., Nielson, C., Osborne, T.F., Raman, R., Ramasamy, K., Sayres, R., Schrouff, J., Seneviratne, M., Sequeira, S., Suresh, H., Veitch, V., Vladymyrov, M., Wang, X., Webster, K., Yadlowsky, S., Yun, T., Zhai, X., Sculley, D.: Underspecification presents challenges for credibility in modern machine learning. CoRR (2020), http://arxiv.org/abs/2011.03395v1
9. Daoud, J.: Animal shelter dataset (2021), version 1. Retrieved March 13, 2021 from https://www.kaggle.com/jackdaoud/animal-shelter-analytics
10. Goyal, A., Lamb, A., Hoffmann, J., Sodhani, S., Levine, S., Bengio, Y., Schölkopf, B.: Recurrent independent mechanisms. arXiv preprint arXiv:1909.10893 (2019)
11. Gulrajani, I., Lopez-Paz, D.: In search of lost domain generalization. arXiv preprint arXiv:2007.01434 (2020)
12. Hu, W., Niu, G., Sato, I., Sugiyama, M.: Does distributionally robust supervised learning give robust classifiers? In: International Conference on Machine Learning. pp. 2029–2037. PMLR (2018)
13. Karimi, K., Hamilton, H.J.: Generation and interpretation of temporal decision rules. arXiv preprint arXiv:1004.3334 (2010)
14. Karimi, K., Hamilton, H.J.: Temporal Rules and Temporal Decision Trees: A C4. 5 Approach. Department of Computer Science, University of Regina Regina, Saskatchewan ... (2001)
15. Ke, N.R., Bilaniuk, O., Goyal, A., Bauer, S., Larochelle, H., Schölkopf, B., Mozer, M.C., Pal, C., Bengio, Y.: Learning neural causal models from unknown interventions. arXiv preprint arXiv:1910.01075 (2019)
16. Locatello, F., Poole, B., Rätsch, G., Schölkopf, B., Bachem, O., Tschannen, M.: Weakly-supervised disentanglement without compromises. In: International Conference on Machine Learning. pp. 6348–6359. PMLR (2020)
17. Mitchell, T.M., et al.: Machine learning (1997)
18. Moneda, L.: Globo esporte news dataset (2020), version 11. Retrieved March 31, 2021 from https://www.kaggle.com/lgmoneda/ge-soccer-clubs-news
19. Mouillé, M.: Kickstarter projects dataset (2018), version 7. Retrieved March 13, 2021 from https://www.kaggle.com/kemical/kickstarter-projects?select=ks-projects-201612.csv
20. Pearl, J.: Causality. Cambridge University Press, Cambridge, UK, 2 edn. (2009). https://doi.org/10.1017/CBO9780511803161
21. Pearson, K.: On a form of spurious correlation which may arise when indices are useed in the measurement of organs. In: Royal Soc., London, Proc. vol. 60, pp. 489–502 (1897)
22. Peters, J., Bühlmann, P., Meinshausen, N.: Causal inference using invariant prediction: identification and confidence intervals. arXiv preprint arXiv:1501.01332 (2015)
23. Peters, J., Janzing, D., Schlkopf, B.: Elements of Causal Inference: Foundations and Learning Algorithms. The MIT Press (2017)
24. Rabanser, S., Günnemann, S., Lipton, Z.C.: Failing loudly: An empirical study of methods for detecting dataset shift (2018)

25. Ribeiro, M.T., Singh, S., Guestrin, C.: " why should i trust you?" explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. pp. 1135–1144 (2016)
26. Schölkopf, B., Janzing, D., Peters, J., Sgouritsa, E., Zhang, K., Mooij, J.: On causal and anticausal learning. arXiv preprint arXiv:1206.6471 (2012)
27. Shastry, A.: San francisco building permits dataset (2018), version 1. Retrieved March 13, 2021 from https://www.kaggle.com/aparnashastry/building-permit-applications-data
28. Sionek, A.: Brazilian e-commerce public dataset by olist (2019), version 7. Retrieved March 13, 2021 from https://www.kaggle.com/olistbr/brazilian-ecommerce
29. Stone, M.: Cross-validatory choice and assessment of statistical predictions. Journal of the Royal Statistical Society: Series B (Methodological) **36**(2), 111–133 (1974)
30. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. arXiv preprint arXiv:1706.03762 (2017)
31. Wager, S., Athey, S.: Estimation and inference of heterogeneous treatment effects using random forests. Journal of the American Statistical Association **113**(523), 1228–1242 (2018)
32. Wilson, A.C., Roelofs, R., Stern, M., Srebro, N., Recht, B.: The marginal value of adaptive gradient methods in machine learning. arXiv preprint arXiv:1705.08292 (2017)