# ADESIT: Visualize the Limits of your Data in a Machine Learning Process

Pierre Faure-Giovagnoli, Marie Le Guilly, Jean-Marc Petit, Vasile-Marian Scuturici

# ADESIT: Visualize the Limits of your Data in a Machine Learning Process

Pierre Faure--Giovagnoli[1,2]
[1]Univ Lyon, INSA Lyon, CNRS, UCBL
LIRIS UMR 5205, Villeurbanne, France
[2]Compagnie Nationale du Rhône
Lyon, France
pierre.faure--giovagnoli@liris.cnrs.fr

Marie Le Guilly
Jean-Marc Petit
Vasile-Marian Scuturici
Univ Lyon, INSA Lyon, CNRS, UCBL
LIRIS UMR 5205, Villeurbanne, France
fisrname.lastname@liris.cnrs.fr

## ABSTRACT

Thanks to the numerous machine learning tools available to us nowadays, it is easier than ever to derive a model from a dataset in the frame of a supervised learning problem. However, when this model behaves poorly compared with an expected performance, the underlying question of the existence of such a model is often underlooked and one might just be tempted to try different parameters or just choose another model architecture. This is why the quality of the learning examples should be considered as early as possible as it acts as a go/no go signal for the following potentially costly learning process. With ADESIT, we provide a way to evaluate the ability of a dataset to perform well for a given supervised learning problem through statistics and visual exploration. Notably, we base our work on recent studies proposing the use of functional dependencies and specifically counterexample analysis to provide dataset cleanliness statistics but also a theoretical upper bound on the prediction accuracy directly linked to the problem settings (measurement uncertainty, expected generalization...). In brief, ADESIT is intended to be part of an iterative data refinement process right after data selection and right before the machine learning process itself. With further analysis for a given problem, the user can characterize, clean and export dynamically selected subsets, allowing to better understand what regions of the data could be refined and where the data precision must be improved by using, for example, new or more precise sensors.

**PVLDB Reference Format:**
Pierre Faure--Giovagnoli, Marie Le Guilly, Jean-Marc Petit and, Vasile-Marian Scuturici. ADESIT: Visualize the Limits of your Data in a Machine Learning Process. PVLDB, 14(12): 2679 - 2682, 2021.
doi:10.14778/3476311.3476318

## 1 INTRODUCTION

We consider a supervised learning (SL) problem where the task is to predict a continuous or categorical target $C$ from a set of features $X$ using a set of examples. The goal is to find a function $f$ (a.k.a. model) such that $f(X) \simeq C$. This function must balance between error minimization on the provided examples and its ability to generalize well on unseen instances to avoid overfitting. To do so, a set of training examples is sent to a learning algorithm to infer such a function which can be evaluated afterwards against a testing dataset. Suppose the resulting accuracy is below expectations: what should be questioned? One might be tempted to try different training parameters or even change the learning algorithm. If this approach makes sense in some cases, challenging the very existence of $f$ should also be one of the primary concerns to prevent an unsuccessful SL process or, conversely, to help the practitioners trust the computed model. In practice, the quality and completeness of the learning examples is one of the primary factors of success in SL. Noisy, inaccurate, or incomplete data can lead to poor models, and it is sometimes even difficult to understand that the learning dataset itself is to blame. Indeed, if the examples given for training contradict themselves or do not contain sufficient information, even the most advanced algorithm will fail at understanding the workings of the phenomenon one wants to predict.

In response to these concerns, multiple methods and metrics have been developed in the past allowing to evaluate, improve, and better understand the data and its predictive power. In particular, we focus on the recent studies proposing the use of functional dependencies (FDs) and specifically the analysis of counterexamples to find contradictions in the dataset [3, 12]. For the specific case of SL, we understand intuitively that learning examples with equal causes ($X$) and different outcomes ($C$) are likely to cause problems during the learning process. As a function, $f$ needs to give a unique answer for a given input. This type of contradiction can be due to noise in the data (sensor precision, input error...) but it might also reveal missing features or just the unpredictable nature of the phenomenon. Therefore, finding regions with high densities of counterexamples is of great interest for data preprocessing, to gain insight into the intrinsic limitations of the raw dataset and the problem statement itself but also to guide interactions with domain experts. If the limits given by counterexample analysis are not suitable for the problem at hand, the process must be refined by working on data acquisition or processing.

As our main proposition, we present ADESIT (Advanced Data Exploration and Selection Interactive Tool), a web-based intuitive graphical user interface to evaluate the limits of a dataset for a given SL problem. This evaluation is made through statistical measures based on counterexamples and an interactive visual exploration permits the user to see the variations of this measure on different regions of the data. Given a dataset, a set of features $X$ and a prediction target $C$, ADESIT helps the user to understand the


Proceedings of the VLDB Endowment, Vol. 14, No. 12 ISSN 2150-8097.
doi:10.14778/3476311.3476318

predictive power of the data but also potential refinements and improvements in her features or data selection. We propose various statistics on the dataset up to tuple granularity along with an interactive representation of the data and found counterexamples. As shown in Figure 1, we thought ADESIT as being part of an iterative refinement process. Given a new SL problem, we want the user to quickly be able to evaluate what performances she cannot exceed and what steps should be taken before approaching the learning process itself.

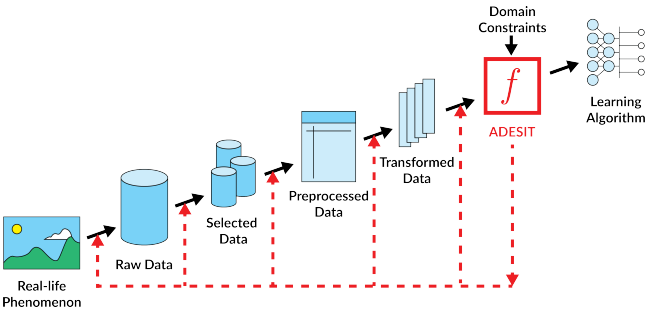


**Figure 1: Place of ADESIT in the data processing pipeline for machine learning. Just before the model creation step, ADESIT allows the user to understand the limits of her dataset, thus taking part of an iterative data refinement process from data acquisition to preprocessing.**

Due to a technical gap between data scientists and experts, domain knowledge is often underused in the conception of prediction models. However, it is often the domain experts themselves who know the intrinsic theoretical parameters of the prediction problem but want a more practical model closer to reality. As a consequence, one of our primary concerns while designing ADESIT was to involve domain experts in the process: the intuitive visualization of counterexamples proposed by ADESIT allows a quick understanding and explanation of the potential issues which may arise, promoting dialogue within all the actors of the data chain.

In summary, for a given dataset and a specific SL problem, ADESIT proposes the following contributions:

- Statistics derived from counterexamples analysis of the associated FD to characterize the ability of the dataset to perform well on the given problem. Notably, we extend previous indicators [10] with domain knowledge integration and propose their first known efficient implementations. One of those indicators serves as an upper bound on the maximum accuracy obtainable with any model on a given dataset [12].
- Scalable counterexample enumeration along with fast approximate and exact computation of the associated indicators allowing fast analysis of large datasets.
- Customizable counterexamples visualizations (scatter plot, table, graph) and selection up to tuple granularity.
- Ways to find and export regions of data with good learning potential but also subsets which need refinements. Those subsets can also be automatically described using SQL queries.
- An intuitive interface focusing on visualization, allowing an easy collaboration between domain experts and data scientists.

# 2 SYSTEM OVERVIEW

## 2.1 Counterexample analysis

For the sake of simplicity, we consider a relation $r[U]$ with $U$ a set of continuous or categorical attributes and a functional dependency $X \rightarrow C$ with $X \cup \{C\} \subseteq U$. We focus on a target $C$ to predict from a set of features $X$ which can either be a classification or a regression SL problem. In our case study for example, it is possible to use the flow and the elevation of the river ($X = \{flow, elevation\}$) to predict the power produced by a water turbine ($C = \{power\}$). Recent studies [3, 5, 12–14] have shown the interest of assessing the veracity of $X \rightarrow C$ regarding a relation $r$. A direct way to evaluate a dataset in view of a given FD is through the study of *violating pairs* (VPs). In the case of classic (a.k.a. crisp) FDs, a pair of tuples $(u, v)$ is defined as a VP if for all $A \in X$, $u[A] = v[A]$ and $u[C] \neq v[C]$. In other words, a VP can be seen as a contradiction in the data where two same sets of features' values lead to different outcomes for $C$. The terms *counterexample* and *violating pair* will be used interchangeably from now on.

[10] proposes three measures to evaluate how a relation $r$ *deviates* from a given FD: $g_1$ measures the ratio of VPs, $g_2$ measures the ratio of tuples involved in a VP and $g_3$ gives the normalized minimum number of tuples one has to delete to obtain a relation without VPs. While $g_1$ and $g_2$ propose evaluations of the dataset purity and counterexamples distribution, $g_3$ provides an appreciation of learnability by proposing an upper bound on the maximum accuracy obtainable with any model on a given dataset [12]. Intuitively, this is true because a model cannot correctly classify a tuple and its associated counterexample at the same time. In the case of crisp FDs, the $g_3$ error can be specifically expressed as follows:

$$g_3(X \rightarrow C, r) = 1 - \frac{max(|s| : s \subseteq r, s \models X \rightarrow C)}{|r|}$$

where $|r|$ is the number of tuples in $r$ and $s \models X \rightarrow C$ means that $X \rightarrow C$ is satisfied in $s$ (i.e. no pair of tuples in $s$ is a VP). However, this definition of VP is known to be too strict as it lacks some crucial domain knowledge integration such as data uncertainty (sensor error, noisy acquisition...). Thus, we propose the use of non-crisp FDs to define VPs and therefore generalized version of $g_1$, $g_2$ and $g_3$ inspired by [14] for conditional matching dependencies (CMDs). Many relaxations of crisp FDs have been proposed in literature, see for instance [1] for a survey. To deal with similarity ($\simeq$) instead of crisp equality, we need to define the satisfaction of non-crisp FDs ($\approx$) and therefore a relaxed version of VPs. $r \not\approx X \rightarrow C$ is now satisfied when there is no pair of tuples $(u, v)$ such that for all $A \in X$, $u[A] \simeq v[A]$ and $u[C] \not\simeq v[C]$. We can therefore define a generalised version of $g_3$ using the same notation for simplicity:

$$g_3(X \rightarrow C, r) = 1 - \frac{max(|s| : s \subseteq r, s \not\approx X \rightarrow C)}{|r|}$$

This generalized $g_3$ indicator extends the initial $g_3$ to propose an upper bound on the accuracy of any model paired with domain knowledge and similarity integration. Among the many similarity measures presented in literature, we consider the following one based on measurement uncertainties: for values $a$ and $b$ of some numerical attribute $A \in U$, $a$ and $b$ are considered as similar under

an absolute threshold $\tau_a$ and a relative threshold $\tau_r$ if:

$$|a - b| \leq \tau_a + \tau_r \times max(|a|, |b|)$$

This similarity measure is notably appropriate for the sensor data used in the demonstration. Note that when $\tau_a$ and $\tau_r$ tend to zero, this definition comes back to that of crisp FDs. Moreover, strict equality is used for categorical attributes but appropriate distance measures (eg. edit distance for textual attributes) as well as other similarity measures could be introduced in the future.

## 2.2 Computation overview

*2.2.1 Violating pairs enumeration.* To enumerate VPs, every tuple has to be compared to each other and two tuples are added to the set of VPs if they agree on X and disagree on C in regard to the similarity measures defined for each attribute. As this operation is quadratic in the number of tuples, it does not scale easily. Nonetheless, such couple enumeration has been extensively studied for *record linkage* and *similarity joins* where optimisations have been proposed for massive datasets. Depending of the attribute type, techniques such as blocking [2] and/or a sliding window algorithm (similar to [4]) can be used for remarkable gains in reducing the processing time. Note that this approach is also highly scalable as many distributed approaches such as MapReduce [11] are presented in literature and could be easily implemented on top of the pipeline.

*2.2.2 Generalized $g_1$, $g_2$ and $g_3$ computation.* Once the VPs have been found, $g_1$ is just the number of VPs and $g_2$ the number of tuples involved in a VP. On the other hand, $g_3$ requires finding the minimum number of tuples to remove from the relation $r$ to satisfy $X \rightarrow C$. If its computation is trivial in the case of classic FDs [9], the use of non-crisp FDs makes it NP-Hard which can be proven by a reduction from the MINIMUM VERTEX COVER (MVC) similar to the one proposed in [14] for CMDs. Moreover, using a graph representation where nodes correspond to tuples and edges connect nodes involved in a VP, computing $g_3$ becomes equivalent to solving the MVC, allowing to benefit from the extensive literature on the subject. To spare the user the exponential complexity of exact solving, we propose an *approximation algorithm* giving upper and lower bounds on the size $|MVC|$ of the MVC. The best-known constant-factor approximation algorithm for the $|MVC|$ upper bound $|MVC|_{ub}$ is described in [7] while the lower bound $|MVC|_{lb}$ is given by the size of maximum matching, such that:

$$0.5|MVC| \leq |MVC|_{lb} \leq |MVC| \leq |MVC|_{ub} \leq 2|MVC|$$

In addition, we also propose an *exact algorithm* which combines multiple graph techniques (kernelization, branch-and-reduce and intermediate local search solutions...) [8] at the price of exponential complexity in the number of edges, and thus only usable for datasets with a reasonable number of counterexamples.

To the best of our knowledge, ADESIT implements for the first time an efficient computation of the generalized $g_3$ measure, both exact and approximated, pointing out an interesting cross-fertilization from database core technologies to machine learning.

## 2.3 ADESIT

ADESIT is a web-based application which proposes a machine-learning-focused analysis and visualization of a given dataset based on counterexample analysis and especially the $g_1$, $g_2$ and $g_3$ measures. Notably, it provides a direct appreciation of the performances a prediction model cannot exceed and ways to increase the quality of input data and improve learning results by observing what tuples do not respect the FD associated to the SL problem at hand. As labelled in the screenshot available in Figure 2, ADESIT's interface is divided into three main areas:

**A** This area is used to define the SL problem. The user first uploads her dataset and can then select attributes to define the features $X$ and the target $C$ of the prediction problem. In conjunction with domain experts, similarity measures based on absolute and relative thresholds need to be defined for each continuous attribute. Finally, the user can choose the type of computation for $g_3$ (approximate or exact) and start analysis (VPs discovery and indicators computation). At the present time, unknown values are not explicitly handled by ADESIT but approaches such as [15] could be implemented in the future.

**B** This area displays the number of tuples involved in a counterexample (unnormalized $g_2$) along with the $g_1$, $g_2$ and $g_3$ indicators. Those indicators are inverted versions of the raw measures (e.g. $g_1$ becomes $1 - g_1$) for readability: *the higher the better*. If the performances are not appropriate for the domain experts, we can intervene on the data collection process by exploring solutions such as increasing data accuracy to influence similarity measures (eg. using better sensors), adding new features or refine the data processing pipeline (acquisition, denoising, gap filling...).

**C** **(C1)** This area is used to visualize the dataset in the perspective of counterexample analysis trough a scatter plot. Each dot in the graph represents a tuple of the dataset and counterexamples are highlighted according to the chosen parameters. By hovering a dot, you can show its specific information and by clicking it, all

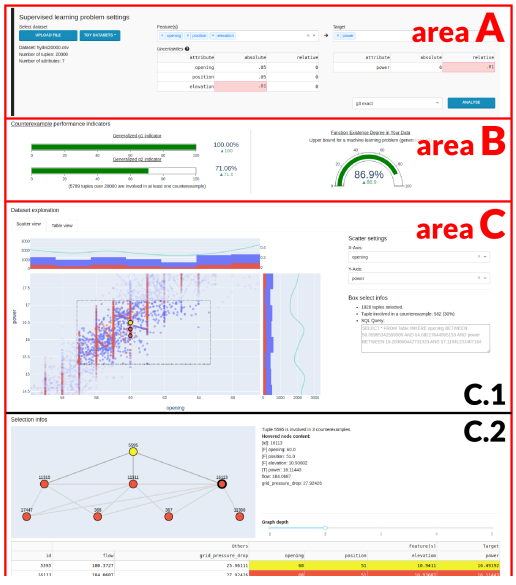


**Figure 2: Labelled screenshot of the ADESIT interface. A: Supervised learning problem settings. B: Counterexample indicators. C: Counterexamples exploration.**

nodes in contradiction get highlighted. Plots of counterexamples distribution are available for each axis combination also including attributes produced with dimensionality reduction techniques such as PCA. In addition to the ability to zoom, move and select tuple regions using lasso or square tools, the user can choose to filter all tuples, counterexamples, or pure tuples only. This visualization allows to detect areas with high densities of counterexamples and helps to decide if the process of data selection for SL can continue or if new or better data is needed. An alternative mirroring table view can also be used to explore and export parts of the dataset. **(C2)** This last area proposes an interactive counterexample graph of variable depth. Nodes can be hovered to get more information on their corresponding tuples and clicked to be selected. A table summarizing the selected tuple and its direct neighborhood is also available.

It is important to emphasize that we are upstream from the model creation itself. The $g_3$ indicator acts as an upper-bound on the target accuracy but will not necessarily be the desired one as it might result in a lack of generalization. However, by making informed choices for the similarity measures (thresholds) and therefore assuming imperfections in the accuracy of the attributes, we produce a better reflection of the real-life phenomenon behind the data and therefore reduce the overall variance. ADESIT is used to identify contexts where creating a good model will be difficult or promising and provides you with knowledge to explain why to potentially refine your data from acquisition (*eg. Should we get new sensors or more precise ones?*) to preprocessing. Also note that ADESIT proposes intuitive indicators *in view of a given dataset* and is not meant to analyze the unpredictability of the underlying phenomenon itself which is notably linked to the Bayes error [6].

ADESIT is coded in Python using the Dash framework. We also use Pandas and Numpy. A beta version of ADESIT hosted on the LIRIS laboratory servers is available at adesit.datavalor.com.

## 3 DEMONSTRATION SCENARIO

For our demonstration, we propose to analyse a real case issue proposed by the Compagnie Nationale du Rhône (CNR), a hydropower company in France. They have recorded various sensor data of one of their hydropower plant over the past ten years at frequencies up to the second and commissioned the following study from us: Is it possible to predict the power produced by a turbine with a precision of 1%? Notably, the given sensor features are rather imprecise with uncertainties up to 5%. Rather than giving them a questionable cryptic number obtained by the costly task of training a lot of different models to get the best accuracy possible, we propose the alternative of ADESIT, allowing to get a direct visual understanding of the limitations of the dataset, facilitating the dialogue with domain experts. Moreover, ADESIT proposes numeric indicators allowing to also give them tangible numbers and notably a theoretical upper bound on the accuracy for their dataset.

The analysis will be deeply linked to domain knowledge as we will use the information provided by the company such as validity domains or uncertainties. We will work with 5 attributes: the power produced (Megawatts) which is the target C we want to predict, the flow of the river ($m^3s^{-1}$), the turbine's blades position (percents), the opening of the water supply (percents) and the elevation of the waterfall (m). We will play with those attributes to show the audience their respective usefulness in the SL problem at hand. Notably, we identify specific areas with more counterexamples which need to be studied closely for training. On Figure 2 for example, we observe that multiple counterexamples occur for *openings* between 50% and 60%. After correlation with the operating curve of the plant and closer examination with experts from the CNR, this observation helped us identify situations of underproduction which could be corrected. Similarly, a range with high density of counterexamples across the *blades position* attribute helped us identify the degradation in time of a turbine. We will take advantage of the complexity of the problem to showcase the diverse functionalities of ADESIT.

During our demonstration, the audience will play the role of the company and we will explain to them in detail what might and might not work with their project. We will notably encourage dialogue and questioning by giving them clear information about the data and the problematic. In addition, we will cover some datasets from the UCI Machine Learning Repository and link our results to the actual performances obtained by the community. Visitors will also be invited to upload their own data to play with ADESIT.

## ACKNOWLEDGMENTS

We thank Matteo Dumont and Antoine Mandin for their help on the initial development of ADESIT. We also thank Benjamin Bertin and Vincent Barellon for testing the application and their help for its deployment. Finally, we thank the Datavalor initiative at INSA Lyon and the CNR for funding a part of this work.

## REFERENCES

[1] Loredana Caruccio, Vincenzo Deufemia, and Giuseppe Polese. 2015. Relaxed functional dependencies—a survey of approaches. *IEEE Transactions on Knowledge and Data Engineering* 28, 1 (2015), 147–165.
[2] Vassilis Christophides, Vasilis Efthymiou, Themis Palpanas, George Papadakis, and Kostas Stefanidis. 2019. End-to-end entity resolution for big data: A survey. *arXiv preprint arXiv:1905.06397* (2019).
[3] Xu Chu, Ihab F Ilyas, and Paolo Papotti. 2013. Discovering denial constraints. *Proceedings of the VLDB Endowment* 6, 13 (2013), 1498–1509.
[4] Vlastislav Dohnal, Claudio Gennaro, and Pavel Zezula. 2003. Similarity join in metric spaces using ed-index. In *International Conference on Database and Expert Systems Applications*. Springer, 484–493.
[5] Wenfei Fan. 2008. Dependencies Revisited for Improving Data Quality *(PODS '08)*. Association for Computing Machinery, New York, NY, USA, 159–170.
[6] Keinosuke Fukunaga. 2013. *Introduction to statistical pattern recognition*. Elsevier.
[7] Michael R Garey and David S Johnson. 1979. *Computers and intractability*. Vol. 174. freeman San Francisco.
[8] Demian Hespe, Sebastian Lamm, Christian Schulz, and Darren Strash. 2020. WeGotYouCovered: The Winning Solver from the PACE 2019 Challenge. In *2020 Proceedings of the SIAM Workshop on Combinatorial Scientific Computing*. SIAM.
[9] Yka Huhtala, Juha Kärkkäinen, Pasi Porkka, and Hannu Toivonen. 1999. TANE: An efficient algorithm for discovering functional and approximate dependencies. *The computer journal* 42, 2 (1999), 100–111.
[10] Jyrki Kivinen and Heikki Mannila. 1995. Approximate inference of functional dependencies from relations. *Theoretical Computer Science* 149, 1 (1995), 129 – 149. Fourth International Conference on Database Theory (ICDT '92).
[11] Lars Kolb, Andreas Thor, and Erhard Rahm. 2010. Parallel Sorted Neighborhood Blocking with MapReduce. arXiv:1010.3053 [cs.DC]
[12] Marie Le Guilly, Jean-Marc Petit, and Vasile-Marian Scuturici. 2020. Evaluating Classification Feasibility Using Functional Dependencies. *Trans. Large Scale Data Knowl. Centered Syst.* 44 (2020), 132–159.
[13] Erhard Rahm and Hong Hai Do. 2000. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.* 23, 4 (2000), 3–13.
[14] Yihan Wang, Shaoxu Song, Lei Chen, Jeffrey Xu Yu, and Hong Cheng. 2017. Discovering conditional matching rules. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 11, 4 (2017), 1–38.
[15] Ziheng Wei and Sebastian Link. 2019. Embedded functional dependencies and data-completeness tailored database design. *Proceedings of the VLDB Endowment* 12, 11 (2019), 1458–1470.