Towards an Inductive Logic Programming Approach for Explaining Black-Box Preference Learning Systems

Fabio A. D’Asaro\textsuperscript{1,3}, Matteo Spezialetti\textsuperscript{1,2}, Luca Raggioli\textsuperscript{1,3}, Silvia Rossi\textsuperscript{1}
\textsuperscript{1}University of Naples Federico II
\textsuperscript{2}University of L’Aquila
\textsuperscript{3}CRdC Tecnologie
matteo.spezialetti@univaq.it, silrossi@unina.it

Abstract

In this paper we advocate the use of Inductive Logic Programming as a device for explaining black-box models, e.g. Support Vector Machines (SVMs), when they are used to learn user preferences. We present a case study where we use the ILP system ILASP to explain the output of SVM classifiers trained on preference datasets. Explanations are produced in terms of weak constraints, which can be easily understood by humans. We use ILASP both as a global and a local approximator for SVMs, score its fidelity, and discuss how its output can prove useful e.g. for interactive learning tasks and for identifying unwanted biases when the original dataset is not available. Finally, we highlight directions for further work and discuss relevant application areas.

1 Introduction

In recent years, there has been a sharp increase of Machine Learning algorithms that are able to match or even exceed human performance in a variety of tasks. Although they are able to carry out complex tasks, these reasoning systems are often regarded as black-boxes, whose internal structure and functioning cannot be easily translated to human-understandable sentences. This issue has raised concern about the applicability, transparency and reproducibility of these black-box models to scenarios in which taking bad decisions can lead to severe consequences, most notably in medical applications, where no trade-offs between accuracy of the model and its interpretability are possible (He et al. 2019). For this reason, the General Data Protection Regulation (GDPR)\textsuperscript{1} introduced the right to an explanation, which aims at guaranteeing the right to obtain meaningful explanations of the logic involved by an automated system to all the individuals potentially affected by its decisions. This much debated principle has provided strong rationale not only to develop transparent Machine Learning algorithms (i.e., algorithms that, by design, can provide explanations for their own decisions), but also to post-hoc methods that automatically extract the logic behind decisions taken by black-box models (see (Guidotti et al. 2018) for a survey of such methods). Transparent systems trade-off between accuracy and ability to explain. On the other hand, post-hoc methods do not alter the black-boxes’ accuracy, and seek to reconstruct explanations for their output that aim to be as clear and correct as possible. In this work we follow the latter approach and propose a post-hoc method to explain black-box models for preference learning using Answer Set Programming (ASP) (Calimeri et al. 2020) and the state-of-the-art Inductive Logic Programming framework ILASP (Law, Russo, and Broda 2014; Law, Russo, and Broda 2015; Law, Russo, and Broda 2018). The standard syntax of ASP defines complex constructs such as weak constraints (Calimeri et al. 2020), that can be used to naturally express preference relations. In turn, these preferences can be learned by ILASP. Thanks to its ability to generalise from examples, the output of any black-box can be processed by ILASP in order to find an appropriate ASP program that mimics its behaviour. There are many advantages to this procedure: first, ASP programs can be readily translated into natural language that can be understood even by non-experts. Second, it can be used to overcome a major issue of non-transparent Machine Learning models: these are in fact often trained on enormous amounts of data which may encode unwanted (moral, racial) biases or artefacts (see e.g. (Caliskan, Bryson, and Narayanan 2017; Schramowski et al. 2020)). In situations where only the black-box model is available (and not the data it was trained on) logic-based method may expose such systematic biases by e.g. showing that decisions are being taken according to prejudices or unfair principles.

2 Background and Related Work

In this section we briefly overview related work, and demonstrate basic definitions from Explainable AI (XAI), Answer Set Programming (ASP) and Inductive Logic Programming (ILP).

XAI is a quickly growing field of study (see e.g. (Guidotti et al. 2018) for a survey) that aims to generate human-understandable explanations for decisions taken by black-boxes. However, little attention has been paid so far to ILP approaches within the field of XAI. Some examples are (Rabold, Siebers, and Schmid 2018; Rabold et al. 2020) and (Shakerin and Gupta 2019). These works, similarly to ours, demonstrate the use of an ILP framework for local explanation of black-boxes using a modification of LIME (Ribeiro, Singh, and Guestrin 2016). Unlike our approach, these works use the Prolog-based system Aleph (Srinivasan \textsuperscript{1}https://eur-lex.europa.eu/eli/reg/2016/679/oj

855
2004) as their ILP system. Our work, instead, is based on the ILP system ILASP which has been proven to scale favourably with respect to Aleph in terms of accuracy in a variety of tasks (Law, Russo, and Broda 2018). A further major difference is that our approach focuses specifically on ranking and preference learning tasks, in which one typically wants to predict an order relation on a collection of objects, e.g. “The user prefers fish over vegetables”. It is worth noting here that ILP is only one of the possible approaches to generate explanations — other methods such as Exceptional Preferences Mining (Rebelo de Sá et al. 2016) and LIME (Ribeiro, Singh, and Guestrin 2016) are possible. We intend to investigate their relationships with ILASP in forthcoming work.

Our work is based on model-theoretic logic programming language ASP. As we previously mentioned, a feature of ASP that is central to our work is that it can represent weak constraints. We briefly show their general form, and then provide the intuition behind their semantics through an example.

Weak constraints have the form:

\[ \neg \exists \mathbf{b}_1, \ldots, \mathbf{b}_n \left[ \mathbf{w} \mathbf{b}_1, \mathbf{t}_1, \ldots, \mathbf{t}_m \right] \]

for terms \( \mathbf{t}_1, \ldots, \mathbf{t}_m, \mathbf{w}, \mathbf{l} \), literals \( \mathbf{b}_1, \ldots, \mathbf{b}_n \) for \( m \geq 0, n \geq 0 \). Terms \( \mathbf{w} \) and \( \mathbf{l} \) are called weight and level of the constraint, respectively. Terms \( \mathbf{t}_1, \ldots, \mathbf{t}_m \) are used to handle independence among weak constraints\(^2\). Unlike hard constraints, weak constraints do not affect the answer sets of a theory. Instead, they induce a preference relation among them. Intuitively, each answer set satisfying the body of a weak constraint gets a penalty that is proportional to that weak constraint’s weight. For instance, consider the ASP program \( \mathcal{P} \) consisting of the following axioms:

\[
\begin{align*}
p(a) & . \ p(b) . \ p(c) . \\
0\{ \ q(X) \} & \leftarrow p(X) .
\end{align*}
\]

This theory has 8 answer sets, namely: \{\( p(a), p(b), p(c) \), \( p(a), p(b), p(c), q(b) \), \( p(a), p(b), p(c), q(c) \), \( p(a), p(b), p(c), q(a) \), \( p(a), p(b), p(c), q(b) \), \( p(a), p(b), p(c), q(c) \), \( p(a), p(b), q(c) \), \( p(a), p(b), q(b) \), \( p(a), q(c) \), \( p(a), q(b) \), \( q(a) \), \( q(b) \), \( q(c) \)\} and \{\( p(a), p(b), p(c), q(a), q(b), q(c) \)\}. Augmenting \( \mathcal{P} \) with the following weak constraints:

\[
\begin{align*}
\neg \exists q(a) . \ [1 \leq 2, a] \\
\neg \exists q(b) . \ [3 \leq 1, b] \\
\neg \exists q(c) . \ [-1 \leq 2, c]
\end{align*}
\]

results again in the 8 answer sets above, since answer sets are not modified by weak constraints. We denote the resulting augmented theory by \( \mathcal{P}^+ \).

To describe the preference relation induced by these weak constraints, we first introduce the notion of cost of an answer set at some priority level \( \mathbf{l} \). This is defined as the sum of weights at priority level \( \mathbf{l} \) for all the weak constraints such that their bodies are satisfied by the answer set. For instance, answer set \{\( p(a), p(b), p(c), q(a), q(b), q(c) \)\} satisfies all the bodies of weak constraints in \( \mathcal{P}^+ \). Therefore its cost at priority level 2 is 0 (which results from adding the weights of the first and third constraint above), whereas its cost at priority level 1 is 3.

We say that an answer set \( A \) is preferred to an answer set \( B \) (according to theory \( T \)) if the cost of \( A \) is strictly smaller than that of \( B \) at the highest level for which their costs differ. For an ASP theory \( T \) this is written \( A \succ_T B \).

Recall theory \( \mathcal{P}^+ \) above, and consider answer sets \( A_1 = \{ p(a), p(b), p(c) \} \) and \( A_2 = \{ p(a), p(b), p(c), q(a), q(b), q(c) \} \). Note that \( A_1 \) does not satisfy any of the bodies of the weak constraints in \( \mathcal{P}^+ \), therefore its cost is 0 at all priority levels. On the other hand, \( A_2 \) satisfies all the bodies of such weak constraints, therefore its cost at priority level 2 is 0, and its cost at level 1 is 3. Therefore, these two answer sets differ at priority level 1, and we conclude that \( A_1 \succ_{\mathcal{P}^+} A_2 \). It is worth noting that negative weights can be assigned to weak constraints. Therefore, for instance, \{\( p(a), p(b), p(c), q(c), q(b) \)\} \( \succ_{\mathcal{P}^+} \{ p(a), p(b), p(c) \)\}.

Weak constraints can be learned using ILASP (Law, Russo, and Broda 2014). To our knowledge, ILASP is the only ILP system that, to date, can learn weak constraints from examples, i.e. ordered pairs of partial answer sets (Law, Russo, and Broda 2015), and therefore it can naturally be applied to preference learning tasks. It should be noted that this generality comes at an increased computational cost with respect to other ILP languages that do not support them. In a nutshell, the user can provide ILASP with a number of ordered pairs, such that the user always prefers the first element of the pair to the second. ILASP, given some additional background knowledge (the language bias) required to define the hypothesis space, will find a suitable theory (including weak constraints) that covers the examples in the input set of preferences. ILASP also works in the case of noisy data, that is, when not all the examples can be covered. Instead, theories that do not cover some of the examples are given a (user-customisable) penalty by ILASP, which then tries to find the theory in the hypothesis space that is minimally penalised. We use ILASP version 3 which is specifically targeted to noisy learning tasks (Law, Russo, and Broda 2018).

Finally, we make a terminological remark. In Explainable AI terminology (Guidotti et al. 2018), the extent up to which an ILASP output (A) satisfies all the examples \( A \) is called fidelity. In the remainder of this paper, we use and measure fidelity in terms of accuracy scores.

### 3 Explaining Black-boxes With ILASP

Similarly to other work on Machine Learning techniques for small preference datasets (see e.g. (Qomariyah and Kaza- kov 2017; Law, Russo, and Broda 2018)), we demonstrate our proposed approach by applying it to Support Vector Machines (SVMs) trained on the SUSHI preference dataset\(^3\) (Kamishima 2003). The SUSHI preference dataset comprises sushi preferences of 5000 users. Each user was asked

\(^2\)We are not going to discuss this syntax here, since all weak constraints in the remainder of this paper are independent from each other. The interested reader can find the full syntax and semantics in (Calimeri et al. 2020).

\(^3\)http://www.kamishima.net/sushi/
Table 1: Features of the SUSHI dataset. Types “cat” and “cont” correspond to categorical and continuous variables, respectively. Feature “style” indicates whether the sushi is maki (0) or not (1), “major group” indicates whether it is made with seafood (0) or not (1), “minor group” describes the subcategory (aomono, akami, shiromi, tare, clam or shell, squid or crab, roe, other seafood, egg, meat, vegetables), “frequency” indicates how often the sushi is eaten and “frequency2” how frequently it is sold.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Type</th>
<th>Range</th>
<th>ASP Encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>style</td>
<td>cat</td>
<td>{0,1}</td>
<td>maki/nothing</td>
</tr>
<tr>
<td>major group</td>
<td>cat</td>
<td>{0,1}</td>
<td>seafood/nothing</td>
</tr>
<tr>
<td>minor group</td>
<td>cat</td>
<td>{0,...,11}</td>
<td>minor_group()</td>
</tr>
<tr>
<td>oiliness</td>
<td>cont</td>
<td>[0,4]</td>
<td>value(oil,·)</td>
</tr>
<tr>
<td>frequency</td>
<td>cont</td>
<td>[0,3]</td>
<td>value(freq,·)</td>
</tr>
<tr>
<td>price</td>
<td>cont</td>
<td>[0,5]</td>
<td>value(price,·)</td>
</tr>
<tr>
<td>frequency2</td>
<td>cont</td>
<td>[0,1]</td>
<td>value(freq2,·)</td>
</tr>
</tbody>
</table>

ILASP as global approximator: In the case of global approximation, one is concerned with finding a transparent representation of the whole black-box, i.e., the logic that may lead to every possible outcome or decision. The procedure for using ILASP as a global approximator of a preference learning system is given in Algorithm 1. It samples \( N \) pairs of items at random from the feature space. Each pair \((i_1, i_2)\) is collected in a list \( \mathcal{O} \) together with the black-box’s prediction \( B(i_1, i_2) \). Note that, thanks to the information recorded in \( B(i_1, i_2) \), the set \( \mathcal{O} \) can be considered as a set ordered pairs (or examples, in the terminology introduced in Section 2) such that the first element of the pair is always preferred to the second element by the underlying black-box. ILASP is then trained on these ordered pairs in \( \mathcal{O} \) using an appropriate language bias \( L \).

An output theory for our example is:

\[
\text{˜} \text{minor_group}(3). \text{[105, 5]} \\
\text{˜} \text{minor_group}(1). \text{[104, 3]}
\]

Algorithm 1: ILASP as a global black-box approximator

Input: Black-box \( B \), language bias \( L \), natural number \( N \)
Output: An ASP theory \( T_B \)

1: \( \mathcal{O} \leftarrow \{} \\
2: \text{for } j \in \{1, 2, \ldots, N\} \text{ do} \\
3: \quad i_1 \leftarrow \text{randomly_sample_from_feature_space()} \\
4: \quad i_2 \leftarrow \text{randomly_sample_from_feature_space()} \\
5: \quad \mathcal{O} \leftarrow \mathcal{O} \cup ((i_1, i_2), B(i_1, i_2)) \\
6: \text{end for} \\
7: T_B \leftarrow \text{ILASP}(L, \mathcal{O}) \\
8: \text{return } T_B
\]

meaning that, according to ILASP, the SVM seems to think that (in order of priority) the User 1 does not like sushis of minor groups 3 and 1, that s/he does not like seafood sushi, that s/he does not like expensive sushis of style maki, and that s/he likes sushi of minor group 9.

The graph on Figure 1 plots the fidelity of ILASP as a function of the number of items it is trained on. Note that, based on a theory, it is not always possible to decide whether a particular type of sushi is preferred over another as different items may get the same penalty score. For example, the output theory above does not allow one to decide which sushi is preferred between two non-maki, non-seafood sushi of minor group 0. We refer to these pairs as unclassified pairs. In Figure 1 we plot both the fidelity for the case where unclassified pairs are regarded as errors, and the fidelity for the case where unclassified pairs are simply discarded and are not considered when computing the fidelity.

ILASP as a local approximator: A local approximator for a black-box \( B \), a query \( Q \) and an associated distance metric \( \pi_Q \) produces an ASP theory \( T_{B,Q} \) that seeks to explain why the black-box \( B \) produces output \( B(Q) \) on input \( Q \). The procedure for using ILASP as a local approximator is given in Algorithm 2 and it is a modification of the LIME method (Ribeiro, Singh, and Guestrin 2016). It departs from Algorithm 1 in that the two items \( i_1 \) and \( i_2 \) are obtained by sampling around \( Q \) (compare lines 3 and 4 of Algorithm 1 with line 3 of Algorithm 2), meaning that pairs that are closer to \( Q \) according to a user-defined distance metric \( \pi_Q \) have a greater probability of being sampled. In addition, these distances are recorded in list \( \mathcal{O} \) and we make ILASP to take them into account during the training phase. Theories that cover examples that are far from \( Q \) are penalised more than theories that cover examples that are close to \( Q \). Pairs of items \((i_1, i_2)\) are weighted by considering a distance metric \( \pi_Q(i_1, i_2) \) to calculate their distance from \( Q \). In our experiments we considered the following metric:

\[
\pi_Q(i_1, i_2) = \sum_{j \in \{1,2\}} \sqrt{\sum_{f} (d(i_j(f), q_j(f)))^2}
\]

where \( i_j(f) \) is the value of feature \( f \) for item \( i_j \), and \( d(i_j(f), q_j(f)) \) evaluates to \( i_j(f) - q_j(f) \) when \( f \) is a con-
a sushi is preferred over another: for instance, the reader may verify that axiom
\[
: \text{value(freq, V0)}.
\]
above does not help differentiating between items \( q_1 \) and \( q_2 \), as it assigns a penalty of 0 to both these sushis.

Note that this does not constitute a limitation: it is rather the case that sometimes too many axioms are being produced by ILASP in the output theory. Therefore, to improve readability, one might want to filter out these constraints and only output those that are relevant to the local explainability task, e.g. one might want to output only the weak constraint
\[
: \text{value(oil, V0)}. \quad [-V0@2, 2, V0]
\]
in this case.

To score the fidelity of our ILASP-based local approximator we performed an experiment where we considered pairs sampled from a neighborhood of \( Q \), both during the training and the testing of ILASP (with disjoint training and test sets). Running Algorithm 2 for the first 10 users on a training set of cardinality 45 for 100 times on random queries resulted in \( \approx 91\% \) fidelity and \( \approx 0.04 \) standard deviation.

### 4 Conclusion

In this work, we performed preliminary experiments on the use of ILP for explaining black-box learning systems. Specifically, we tested the ILASP framework both as global and local approximator of a classic machine learning algorithm (SVM), trained to classify user’s preferences regarding sushi. When employed as global estimator, our approach achieved 75.8% and 78.3% of fidelity considering unclassified data as errors or not taking it in account, respectively. Best results were obtained by training the system with a set of about 80 samples (dichotomic preferences), but, as shown in Figure 1, after 45 samples, fidelity values both converge to a plateau (71.0% - 74.2%). For this reason, we chose 45 as training set size for testing our local approximator. In this case ILASP scored an average fidelity of 91.0%, producing a quite good local explanation for SVM queries. At the moment, fidelity is the only metric we adopt to evaluate our approach. If, in addition, absolute preferences of the users were available (e.g., “User1 rates oily sushis 7 out of 10”) we would be able to validate output theories on them. For this reason, we plan to build a dataset that also includes such information.

![Figure 1: Fidelity of ILASP as a global approximator vs. cardinality of training set](image-url)
Ongoing work includes efforts to incorporate fragments of this framework, alongside machine learning black-boxes and other logic-based systems (D’Asaro et al. 2017; D’Asaro et al. 2020), into a decision-making support system in the context of healthcare project AVATEA (D’Asaro, Origlia, and Rossi 2019). The project requires to take decisions and provide natural language explanations and feedback about users’ preferred rehabilitation strategies. Often, these decision are taken by black-box classifiers, and being able to provide therapists with useful information about them is a non-trivial task that can be addressed using some of the techniques described here.

Beside its explanation power, the proposed method could be useful to discover unwanted biases and artefacts in the training set the black-box was originally trained upon. This could make black-boxes users aware of such biases and enable them to provide feedback on such explanations (e.g., it might be the case that a user disagrees with the decisions taken). This could in principle be used to interact with the user and ask him/her to provide more data if such biases seem to be present. Following this idea, future developments will include the gathering of a large dataset, also featuring meta-information on the user preferences about items’ features.

Acknowledgements

This work was partially supported by the Italian Ministry of University and Research (MIUR) within the POR Campania FESR 2014-2020 AVATEA “Advanced Virtual Adaptive Technologies e-HEAlth” research project, and within the PRIN2015 research project ”UPA4SAR—User-centered Profiling and Adaptation for Socially Assistive Robotics” (grant no. 2015KB-L78T).

References


