proceedings

Proceedings

aspBEEF: Explaining Predictions Through Optimal Clustering †

Pedro Cabalar [1], Rodrigo Martín [1], Brais Muñiz [1,2,*] and Gilberto Pérez [1,2]

1 Department of Computer Science, University of A Coruña, 15001 A Coruña, Spain; pedro.cabalar@udc.es (P.C.); r.martin1@udc.es (R.M.); gilberto.pvega@udc.es (G.P.)
2 CITIC Research Center, University of A Coruña, 15001 A Coruña, Spain
* Correspondence: brais.mcastro@udc.es
† Presented at the 3rd XoveTIC Conference, A Coruña, Spain, 8–9 October 2020.

Published: 28 August 2020

Abstract: In this paper we introduce aspBEEF, a tool for generating explanations for the outcome of an arbitrary machine learning classifier. This is done using Grover's et al. framework known as Balanced English Explanations of Forecasts (BEEF) that generates explanations in terms of in terms of finite intervals over the values of the input features. Since the problem of obtaining an optimal BEEF explanation has been proved to be NP-complete, BEEF existing implementation computes an approximation. In this work we use instead an encoding into the Answer Set Programming paradigm, specialized in solving NP problems, to guarantee that the computed solutions are optimal.

Keywords: knowledge representation; answer set programming; explainable AI

1. Introduction

Explainability is one of the clear barriers Machine Learning (ML) has not yet overcome. In some domains, such as medicine, where decisions can seriously affect people's lives, experts need to understand the decisions of ML models, to avoid biased or unfair decisions. Unfortunately, explaining why a classifier makes a given prediction is a non-trivial task, specially in human terms. Explainable Artificial Intelligence (XAI) field studies tools that aim to explain the predictions of ML models by externally analyzing them, treating models as black boxes. This approaches are independent of the algorithm's design and would be able to explain already deployed models retrospectively. One example of such a tool is the framework *Balanced English Explanations of Forecasts* (BEEF) [1] which produces natural language explanations of an ML model by grouping theirs predictions into clusters. However, the BEEF process of finding such clusters is heuristic-guided and does not provide the optimal solutions. In this paper we present a prototype called aspBEEF, which grants the set optimal of BEEF clusters through the use of Answer Set Programming (ASP) [2–4]. In the following text we first explain in more detail how BEEF computes its explanations. Then, we present how our prototype aspBEEF works. After that, we show a short evaluation of the prototype. Finally, we comment about the future work and conclude the paper.

2. BEEF Computation of Clusters

BEEF is a framework to explain the outcome of any ML binary classifier in terms of intervals over the input features. Given a set of predictions, BEEF first clusters them using some traditional method (such as KMeans). The result of the clustering is used as a starting point to find a set of *axis-aligned, hyperrectangular clusters* (boxes for short) that satisfy some previously given thresholds in terms of (1) purity: the majority predicted class in each cluster; (2) overlapping: the amount of space shared by several clusters; and (3) inclusion: the lack of predictions outside any cluster. The algorithm iteratively adjusts the boundaries of the boxes, until the thresholds are satisfied. This problem has been proved to be NP-complete by Grover et al. [1]. Their framework uses some heuristics during the search, at the cost of losing optimality.

Boxes are *axis-aligned* and finite, so they can be described as a set of intervals over each dimension (each input feature). Each box is labeled with the name of the majority predicted class within. We can explain a model outcome *o* just finding out the description (the intervals) of the boxes *o* fell in. The descriptions of those boxes whose label matches the predicted class will be the *supporting explanations*. The rest will be the *opposing explanations*. The sum of supporting and opposing explanations is what BEEF calls a balanced explanation.

3. The `aspBEEF` Tool

For finding the optimal solutions, `aspBEEF` makes use of ASP, a declarative Knowledge Representation and problem-solving paradigm. Under ASP, the domain knowledge and the problem to solve must be specified as a set of rules in a logic program, from which a solver obtain the solutions in terms of models of the program called answer sets. ASP also allows optimization among answer sets in different ways: in this work, we use the ASP extension `asprin` [5] that allows a flexible way of defining preference relations for optimization. In particular, the use of `asprin` preference relations enables us to shift between simple and general explanations or fitted and complex ones depending on the problem.

`aspBEEF` takes the ML model predictions, the number of clusters and the feature configuration to generate the BEEF balanced explanations. The tool provides those explanations as a set of rules in ASP or as a graphical representation. Since development is not closed yet, there exist some differences with BEEF. While BEEF maximizes inclusion, `aspBEEF` minimizes exclusion since ASP handles minimization easier. Furthermore, BEEF clusters have their own set of active dimensions while in `aspBEEF` the dimensions are active or inactive globally. Finally, `aspBEEF` allows the user to choose the number of different features to use, some of those can be fixed to reduce the search space.

4. Evaluation

Evaluation has been done by running `aspBEEF` with random samples of three different sizes of the publicly available IRIS data set (https://archive.ics.uci.edu/ml/datasets/iris) and using both the fixed and free feature selection methods to compare performance. Each measure has been taken 100 times and then averaged to smooth out outlying values, as ASP solving is a non-deterministic algorithm.

The search space and the computational time grows exponentially with the number of free features. Cases in which all of the features are selected, (e.g., all of the four features in the case of Table 1) eliminate any decision-making over feature selection completely, thus greatly reducing the problem complexity. The best times are achieved using a pre-fixed set of features, but this requires previous knowledge of the search space.

Despite the long computational times when using free features, automatic feature selection provides additional insight about the explanations given by the system, as it prunes the less useful features. Nevertheless, performance is acceptable even for the most complex case-scenarios.

Table 1. Times table for a data set of 150 points and 4 features.

Sample Size	Used Features	Time w/ Free Features	Time w/ Fixed Features
60	2	1.1288	0.7253
60	3	1.1103	0.6995
60	4	0.5443	0.5700
90	2	3.4666	1.6238
90	3	2.5741	1.3963
90	4	1.3596	1.1760
150	2	27.7299	5.5057
150	3	28.8644	5.9140
150	4	7.7072	6.1569

5. Conclusions and Future Work

We have introduced the tool `aspBEEF` whose main feature is to obtain optimal explanations for the BEEF framework. Evaluation shows that the technique has a high computational cost. To solve this problem, feature selection should be performed externally, to fix the important features. Anyway the effort is reasonable having in mind that our goal is not simple classification but obtaining rich explanations of the outcome of any ML classifier.

As future work, we aim to implement BEEF feature selection, modifying the activation of features to be cluster-dependent. Furthermore, we want to improve flexibility by adding an option to make `aspBEEF` find the best number or rectangles, instead of being a user's choice.

The `aspBEEF` prototype is open source and already available at https://github.com/Trigork/aspBEEF.

Funding: This work has been partially supported by MINECO (grant TIN2017-84453-P) and Xunta de Galicia (grants GPC ED431B 2019/03 and ED431G 2019/01 for CITIC center).

References

1. Grover, S.; Pulice, C.; Simari, G.I.; Subrahmanian, V.S. BEEF: Balanced English Explanations of Forecasts. *IEEE Trans. Comput. Soc. Syst.* **2019**, *6*, 350–364. doi:10.1109/TCSS.2019.2902490.
2. Niemelä, I. Logic Programs with Stable Model Semantics as a Constraint Programming Paradigm. *Ann. Math. Artif. Intell.* **1999**, *25*, 241–273.
3. Marek, V.W.; Truszczyński, M. Stable Models and an Alternative Logic Programming Paradigm. In *The Logic Programming Paradigm*; Apt, K.R., Marek, V.W., Truszczyński, M., Warren, D., Eds.; Artificial Intelligence, Springer: Berlin/Heidelberg, Germany, 1999; pp. 375–398.
4. Brewka, G.; Eiter, T.; Truszczynski, M. Answer set programming at a glance. *Commun. ACM* **2011**, *54*, 92–103. doi:10.1145/2043174.2043195.
5. Brewka, G.; Delgrande, J.P.; Romero, J.; Schaub, T. asprin: Customizing Answer Set Preferences without a Headache. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015; pp. 1467–1474.

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).