Explainability of High Energy Physics events classification using SHAP

R Pezoa[1,2], **L Salinas**[2,3], **C Torres**[2,3]

[1] Escuela de Informática, Facultad de Ingeniería, Universidad de Valparaíso
[2] Centro Científico Tecnológico de Valparaíso, Universidad Técnica Federico Santa María
[3] Departamento de Informática, Universidad Técnica Federico Santa María

E-mail: raquel.pezoa@uv.cl, lsalinas@inf.utfsm.cl, ctorres@inf.utfsm.cl

Abstract. Complex machine learning models have been fundamental for achieving accurate results regarding events classification in High Energy Physics (HEP). However, these complex models or *black-box* systems lack transparency and interpretability. In this work, we use the SHapley Additive exPlanations (SHAP) method for explaining the output of two event machine learning classifiers, based on eXtreme Gradient Boost (XGBoost) and deep neural networks (DNN). We compute SHAP values to interpret the results and analyze the importance of individual features, and the experiments show that SHAP method has high potential for understanding complex machine learning model in the context of high energy physics.

1. Introduction

Classifying High Energy Physics (HEP) events, or separating *signal* events from the *background*, is one of the most important analysis tasks in the HEP field, and a fundamental work in the research of new phenomena. The complicate nature of HEP processes requires the use of complex Machine Learning (ML) classifiers, like tree ensembles with thousands of trees, or deep neural networks with thousands of layers and millions of parameters [6]. These complex ML models are viewed as *black-box* systems that frequently lack transparency and interpretability, as opposed to the white-box systems like linear or decision trees-based models, that are more simple and understandable, but are less accurate than black-box systems. Fully understanding complex ML models increase their reliability in accurately identifying physics of interest and also in drawing conclusions about proposed theories.

Explainable Artificial Intelligence (XAI) is a current research field that proposes methods and techniques for producing more explainable models, and as well as a conceptual framework for a better understanding of the predictions performed by artificial intelligence systems [4]. This work is focused on the use of the SHapley Additive exPlanations (SHAP) [9] method, a post-hoc explainability technique from the XAI field that creates a new explanation model for providing *local explanation* by assigning to each feature of each data point an importance value on the model's prediction. More precisely, in this work we use the SHAP method for interpreting the predictions performed by two HEP event classifiers that are based on eXtreme Gradient Boost (XGBoost) [5] and deep neural networks (DNN). The classification problem in this paper consists of identifying the Higgs boson, which is the signal of interest, and the goal is to separate it from the background. The public *Higgs* dataset described in [3] is used.

The Python *shap* framework [9] is used in the paper. This framework provides diverse methods for computing the importance features values, as well as for the visualization of the feature contribution on the model's prediction. In addition, by aggregating the local explanations, it is possible to compute *global explanations*, capturing global patterns that contribute to understand the model's behavior.

This paper is organized as follows. In Section 2 we describe the dataset and the classifiers configuration. Then, in Section 3 we introduce the SHAP method and show the SHAP values of each HEP event classifier, using diverse plots provided by the *shap* module. Finally, in Section 4 we present the conclusions and the future work.

2. High energy physics events classification

2.1. Dataset

In this work, we use a publicly available benchmark dataset, corresponding to simulated data generated to perform the event classification task. The dataset is downloaded from OpenML [10], and it is a subset of the *Higgs* dataset described in [3]. Here, the signal process corresponds to the production of new theoretical Higgs bosons, and the background is the process with identical decay products but different kinematic features. More precisely, the signal is the process defined by: $gg \to H^0 \to W^{\mp}H^{\pm} \to W^{\mp}W^{\pm}h^0 \to W^{\mp}W^{\pm}b\bar{b}$, where the signal process starts with the fusion of two gluons (gg), with intermediate Higgs and W bosons that finally decay into a pair of b quarks.

Each event is represented by a feature vector $x \in \mathbb{R}^{28}$ of 21 low-level features corresponding to physics properties measured by the detector, and 7 high level features derived from the previous ones. The low level features consists of transverse momentum measurements (lepton_pT, jet1pT, jet2pt, jet3pt, jet4pt), pseudorapidity (lepton_eta, jet1eta, jet2eta, jet3eta, jet4eta), azimuthal angles (lepton_phi, jet1phi, jet2phi, jet3phi, jet4phi), b-tagging information of each jet (jet1b-tag, jet2b-tag, jet3b-tag, jet4b-tag), missing_energy_magnitude, and missing_energy_phi. The high-level features include reconstructed masses: m_jj, m_jjj, m_lv, m_jlv, m_bb, m_wbb, and m_wwbb. Here, l, v, j, and b are the notations for lepton, neutrino, jet, and b-quark, respectively. In addition, each event has the class label $y \in \{0, 1\}$, and from the machine learning point of view, this a binary classification problem, where events are classified as signals ($y = 1$) or background ($y = 0$).

2.2. Building the classifiers

Machine learning (ML) has been fundamental for the analysis of HEP data [2] and in this work, we build two ML classifiers, using eXtreme Gradient Boost (XGBoost) and deep neural networks (DNN). We train and evaluate each classifier using the *Higgs* dataset which is divided into training, validation and test sets in a 60%, 20%, 20% split. Models were trained in `http://www.hpc.utfsm.cl` cluster. Then, each classifier together with the testing dataset will be the input of the SHAP method for interpreting the outputs of the models, as shown in Figure 1.

2.2.1. XGBoost classifier. XGBoost [5] is a tree based ensemble ML technique with an efficient implementation of gradient boosted algorithm. For model training, we used the Python XGBoost package and the scikit-learn library. Several parameters need to be selected to maximize model performance, hence we performed parameter tuning using a grid search approach, and 10-fold cross-validation, with ROC AUC as selection metric. The best model were obtained for the following parameter values: max_depth = 2, min_child_weight = 5, subsample = 0.5, colsample_bytree = 0.8, eta=0.01, and eval_metric = auc, using the binary logistic as objective function, and n_rounds = 5000. As a result, the best XGBoost classifier achieved F1-score 0.75, precision 0.63, recall 0.93, accuracy 0.67, and AUC 0.81.

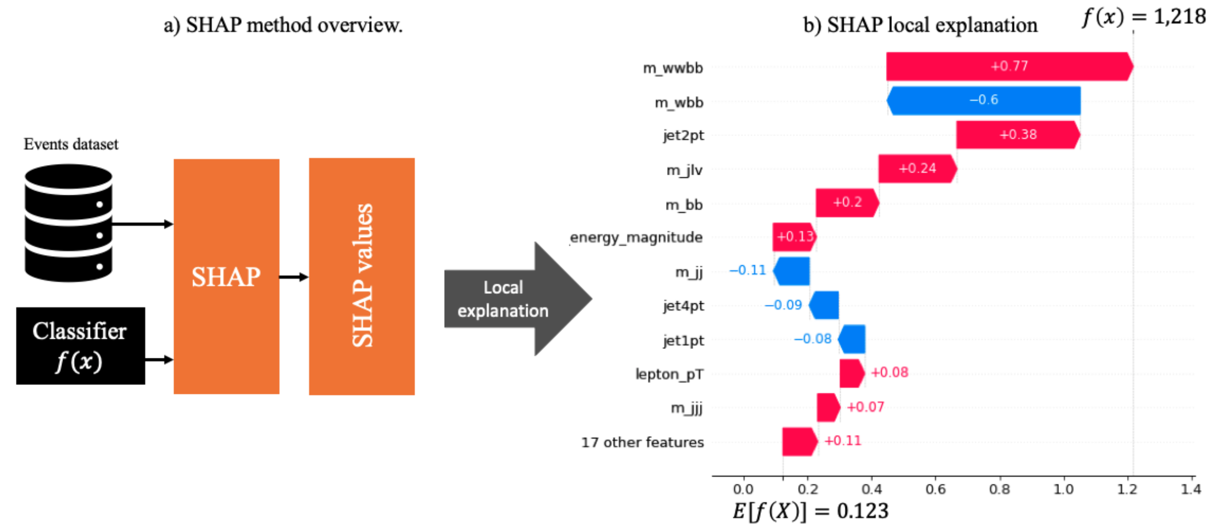


Figure 1. (a) Schematic representation of the local explanation of an event classifier using SHAP. (b) SHAP local explanation using the *waterfall* plot. We observe the SHAP values of an individual event's features. The prediction of the classifier (XGBoost) is $f(x) = 1.218$ and the base value $E[f(x)] = 0.123$. Here, the feature m_wwbb has a SHAP value +0.77 indicating is pushing higher the prediction value, and the feature m_wbb has a SHAP value −0.6, and hence, is pushing lower.

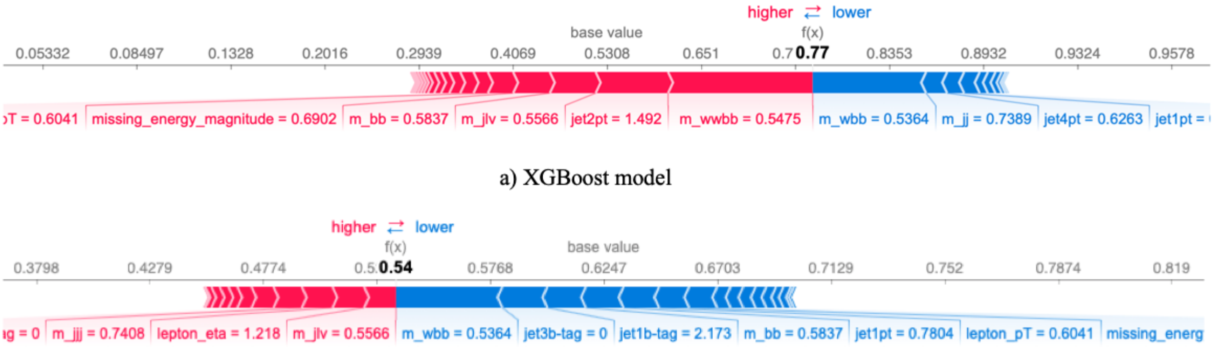
2.2.2. DNN Classifier. Deep learning, i.e., artificial neural networks with multiple layers, is a ML approach that have achieved impressive results in different research fields, including HEP. In this work, we use a fully connected neural network, and the architecture together with the hyper-parameters were found using Talos [1], a Python hyperparameter optimization framework that can be used together with Keras. The best results were obtained for a *brick* shape network with 4 hidden layers with 64 neurons each one, and the ReLu activation function. The output layer has 2 neurons and the sigmoid activation function, as usual in binary classification. We use the binary cross entropy loss function, the AUC validation metric, and the Adam optimizer, with learning rate 1.0, batch size 512, and 100 epochs. The best DNN classifier achieved F1-score 0.67, precision 0.68, recall 0.66, accuracy 0.66, and AUC-ROC 0.71.

3. Explainability using SHAP

3.1. SHAP method

Once the classifiers are built, the next step is to explore the model explainability. The SHapley Additive exPlanations or SHAP [9] method is a technique of the XAI field based on concepts of the cooperative game theory to compute a unified measure of feature importance, the so-called SHAP values, which summarize the importance of each feature on the model prediction. SHAP is a *post-hoc* and model-agnostic method, i.e., it creates a new explanation model for a given black-box system, by extracting relationships between the feature values and the black-box system output [4].

In this work, we interpret the predictions of the HEP event classifiers using the tools provided by SHAP. As shown in Figure 1.(a), the SHAP method receives as inputs the ML classifier (the XGBoost or DNN classifier) and a set of events, and generates the SHAP values of each event of that set. Then, the local explanations can be visualized using the plots provided by the *shap* Python module, like the *waterfall* plot depicted in Figure 1.(b). This plot allows us to visualize



a) XGBoost model

b) DNN model

Figure 2. Force plots. SHAP values of a single prediction of the (a) XGBoost and (b) DNN classifiers. Features that contributes to higher and lower SHAP values are shown in red and blue, respectively, along with the size of each feature's contribution to the model's output.

the SHAP values of each feature in a particular event. Here, the model prediction made by the XGBoost classifier is $f(x) = 1.218$ and the base value (or expected value) is $E[f(x)] = 0.123$. Note that in this work, the classifiers predict an score in the range $[0, 1]$, but in the waterfall plot, the prediction an base values are the values before to transform log odds to probabilities.

As described in [9], SHAP is an *additive feature explanation method* that considers a ML model f that predicts $y = f(x)$ from an input $x = [x_1, ..., x_M]$, and an explanation model g that is defined as a linear function of binary variables, such that $g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i$. Here, $z' \in \{0, 1\}^M$ is a binary vector of ones and zeros: one if the feature is present and zero if not. M is the number of features, and $\phi_i \in \mathbb{R}$ are the feature importance values.

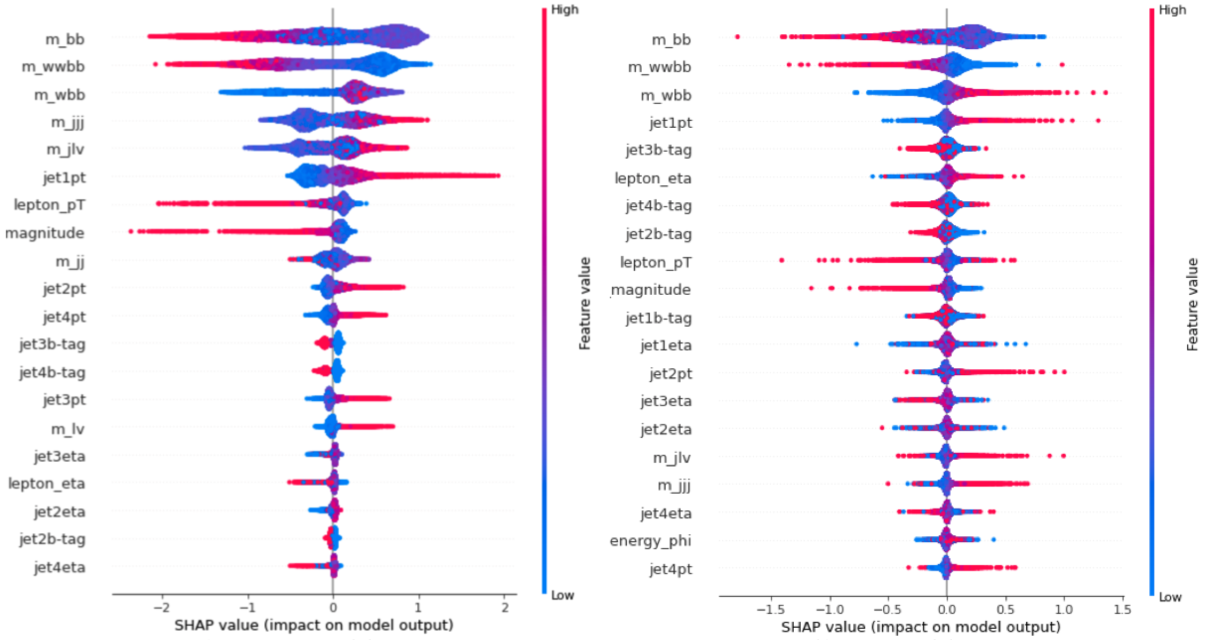
Briefly speaking, the SHAP values are computed by combining the Shapley values (from game theory) and the conditional expectaction function of the original model f. When the explanation model g satisfies the following three properties: local accuracy, missingness, and consistency [9, 7], the SHAP values are given by:

$$\phi_i(f, x) = \sum_{S \subseteq S_{all \setminus \{i\}}} \frac{|S|!(M - |S| - 1)!}{M!} [f_x(S \cup \{i\}) - f_x(S)] \tag{1}$$

where $S_{all \setminus \{i\}}$ is the set of all features, S is the set of non-zero indices in z', and $f_x(S) = E[f(x)|x_S]$ is the expected output of the model conditioned to the feature values of the subset S [7]. The exact determination of SHAP values is computationally expensive, but there are different variants of the method like Tree SHAP algorithm [8] that deals with decision trees-based models and calculates exact SHAP values in a fast way, or the Deep SHAP [9] algorithm which is optimized to explain models based on deep neural networks.

3.2. SHAP values

We use the *TreeExplainer* and *DeepExplainer* methods, provided by the *shap* Python module, and each one implements the Tree SHAP and Deep SHAP algorithms, respectively, for computing the SHAP values. Then, we use *force plots*, which show how each feature *forces* the output value of the model. As shown in Figure 2, the red/blue color indicates that features push the prediction higher/lower, and the magnitude corresponds to the amount of contribution of each feature on the model's prediction. The XGBoost classifier predicts an score in the range $[0, 1]$, and then, by selecting a threshold, the classification is made: the event is classified as signal if the XGBoost



a) XGBoost model b) DNN model

Figure 3. Global explanation. Summary plots for interpret the global feature influences on HEP event classification. m_bb and m_wwbb are the most contributing features on the model's prediction. Each dot represents an event, and a higher SHAP value (x-axis position) indicates the model predicts higher, and vice-versa.

prediction is larger than the threshold and is classified as background otherwise. Hence, in the example of Figure 2, the XGBoost model predicted $f(x) = 0.77$, and the feature m_wwbb whose value is 0.5475 is pushing higher the prediction of the model, i.e., this feature is pushing to predict the signal class, and its contribution is quantified with the SHAP value equal to $+0.77$ (SHAP values of this event are shown in Figure 1.(b)). On the other hand, the feature m_wbb whose value is 0.5364 is pushing the prediction to the background class, with a SHAP value equal to -0.6.

The features contribution when the DNN classifier is used to predict on the same previous event is different compared the the XGBoost model. As shown in Figure 2.(b), the DNN model predicted $f(x) = 0.54$, and the features m_wbb $= 0.5364$, jet3b-tag $= 0$, jet1-btag $= 2.173$, and m_bb $= 0.5837$ with SHAP values equal to -0.19, -0.08, -0.07, and -0.07, respectively (obtained from the computational experiments), are pushing the prediction lower (to background class), and the feature m_jlv $= 0.5566$ is pushing higher, with the SHAP value equal to $+0.06$. Note that this force plot can be obtained for each instance of the testing set, but can also can be obtained for the entire dataset (more plots available at `https://github.com/rpezoa/hep_shap/`).

By combining the local explanations it is also possible to obtain a *global explanation*. The SHAP *summary plot* provides this information, and Figure 3 depicts this plot for each classifier. Here, each dot represents an individual event of the testing set. The y-axis shows a feature rank in descending order (we show the 20 first features), the position of the dots on the x-axis indicates the individual impact (the SHAP values) of each feature, and its gradient color indicates the value of the feature. In the plot of Figure 3 we can observe that the m_bb, m_wwbb, and m_wbb are the most important features on average in both models. We also see that the lower values of m_bb and m_wwbb (blue dots) push prediction higher (higher SHAP values), but in the case of the XGBoost model, this impact is bigger due to the cluster of blue dots is further from the center compared to the DNN model. Hence, values close to the center represent less impact, like jet2b-tag feature. Figure 3 also shows that both classifiers coincide in 17 of the first 20 most important variables, but there is a difference in the order of the feature importance and in the magnitude of impact of each feature. For instance, the XGBoost model has the feature m_jjj in the fourth place of importance, but the DNN model has this feature in the place number 17. This result shows us the different nature of the learning process of each classifier, and hence, the application of an explainability method like SHAP can help us to strategically control and adjust the *black-box* system.

4. Conclusions

We have shown the application of the SHapley Additive exPlanations (SHAP) method to explain the output of two HEP events classifiers, based on eXtreme Gradient Boost (XGBoost) and deep neural networks (DNN), using the *Higgs* public dataset for classifying the signal from background. The present paper shows the application of SHAP method in complex machine learning systems, or *black-box* systems, in the context of HEP events classifiers. Using the *TreeExplainer* and *DeepExplainer* methods, provided by the Python *shap* library, we computed the SHAP values, i.e, we calculated a measure that summarizes the feature contribution of each event on the model prediction. Results showed the top-level features m_bb, m_wwbb, and m_wbb were the most important features on average in both models, contributing to push higher the model's prediction, i.e., the value of $f(x)$ is higher, pushing to predict the signal class. Even though both classifiers obtained a similar set of most important features, showed a different distribution of SHAP values, which clearly shows a distinct learning process on each classifier. This work is a starting point for contributing to a rigorous understanding of physics processes when machine learning methods are used. The future work include the development of a framework able to explain different machine learning models using SHAP, using datasets of other physics phenomena of interest. In addition, SHAP values can be used as a feature selection technique, and a comparison with the traditional feature selection methods will be performed.

5. Acknowledgements

We gratefully acknowledge the financial support from ANID FONDECYT Postdoc Project N. 3190740 and ANID PIA/APOYO AFB180002.

References

[1] Autonomio Talos. `http://github.com/autonomio/talos`, 2019.

[2] Kim Albertsson et al. Machine Learning in High Energy Physics Community White Paper. 2018.

[3] Pierre Baldi, Peter Sadowski, and Daniel Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5(1):1–9, 2014.

[4] Alejandro Barredo Arrieta et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115, 2020.

[5] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[6] Dan Guest, Kyle Cranmer, and Daniel Whiteson. Deep learning and its application to lhc physics. *Annual Review of Nuclear and Particle Science*, 68:161–181, 2018.

[7] Scott M Lundberg et al. Explainable machine-learning predictions for the prevention of hypoxaemia during surgery. *Nature biomedical engineering*, 2(10):749–760, 2018.

[8] Scott M. Lundberg et al. From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, 2(1):2522–5839, 2020.

[9] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777, 2017.

[10] Joaquin Vanschoren, Jan N Van Rijn, Bernd Bischl, and Luis Torgo. OpenML: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60, 2014.