Study of the weak feed-down contamination in Λ baryons using Machine Learning

August 2, 2024

Contents

1 Introduction

The lightest baryon containing strangeness is the Lambda hyperon, which consists of an up- a down- and a strange valence quark. Λ hyperon are produced and observed in heavy ion and pp collisions. They can not be tracked in a detector directly, because they do not have an electric charge, but they decay weakly into a proton and a pion. The tracks of these two particles can be tracked in the Alice detector, which leads to a very characteristic V-shape (the proton and pion propagating in different directions from the same starting point, the decay vertex of the Λ-baryon). A particle whose decay products have this characteristic shape are called V0. Of course, the collision of heavy ions produces countless particles. In the experiment, these make their way through a detector and decay into even more particles, other baryons can arise and therefore new Λ-baryons can arise. If one V0 particle is now detected with the invariant mass of a Λ, it is initially not known whether it is a product of the original collision (which we will refer to as "prompt") or a decay product

("non-prompt"). The neutral Xi baryon primarily decays into a lambda baryon and a neutral pion. Alternatively, it can decay into a Sigma zero baryon and a neutral pion. The Xi minus baryon predominantly decays into a lambda baryon and a negatively charged pion, or it can decay into a Sigma minus baryon and a neutral pion. The fraction (20%) of Λs that come from the feed-down of heavier strange baryon states (as teh mentioned Xi particles) is significant. This fraction has never been measured in a pure data driven way in ALICE.
There are several features of the particles that the Alice detector can measure. The distributions of the Lambda kinematic and topology variables allow in principle to disentangle between prompt and non-prompt production. This problem of disentanglement is difficult to solve. The value that a feature (for example the Distance of closest approach to the primary Vertex, DcaV0PV) of a particle can take on is not limited to a single value, but is distributed in a characteristic distribution. These distributions look differently for prompt and non-prompt particles, but they overlap, so it is not possible to classify a single particle manually. The problem is even more complicated because besides prompt and non-prompt particles, there is a background noise from various processes, and wrongly matched particles. Therefore machine learning algorithms are used to classify the data of the strange particles detected in the Alice detector to prompt, non-prompt and background.

2 Machine learning

Machine learning is often labeled as something completely new and sometimes treated as a magic process. But the main principle of the machine learning is to perform a fit where fitting means using data to make future predictions.
In a basic fit of a physical problem one defines a function $f(x)$ (where x refers to any kinematic variable), that depends on a set of parameters a_k: $f(x) = f(x, a_k)$. The form of f depends on the problem to be solved and the ansatz is therefore made with the knowledge or the physical intuition of the scientists. The parameters a_k can be seen as natural constants, there is no theory available that can describe what value they should have and therefore they have to be determined in the fit. This procedure becomes problematic when there is no possibility to make an educated guess because the problem is to complex f(or example the problem that millions of particles produced in a heavy ion collision fly through a detector). In this case the scientists can not even know the form of the desired function $f(x)$. The main idea of the machine learning is then to construct a model function on a sample to fix the internal parameters, to test the performance of the model by applying it on a test sample. If the performance is good, the model can describe the problem out of available data.

2.1 The structure of machine learning

In this project the data analysis tools from *hipe4ml* [1] are used, and the *xgboost* is chosen as a classification model [2]. The classification of *xgboost* works with binary decision trees, an example can be seen in fig. 1. A data point enters the tree and at every node, it is classified to the right or to the left side. At the end of the tree the data point gets a score and the combination of the scores of all trees for the data point defines its so called BDT-score.
The features to make decisions can be considered as a vector $\vec{x} = \{x_1, x_2, ..., x_n\}$. Each node is a function of a single feature (larger or less then a specific value of this feature), but in one tree various features can be used and should be used to capture complex interactions between the features.

The trees should be fitted with data for which we know its classification which means that the training set is a sample of known nature, for example a Monte Carlo one. The goal is to make

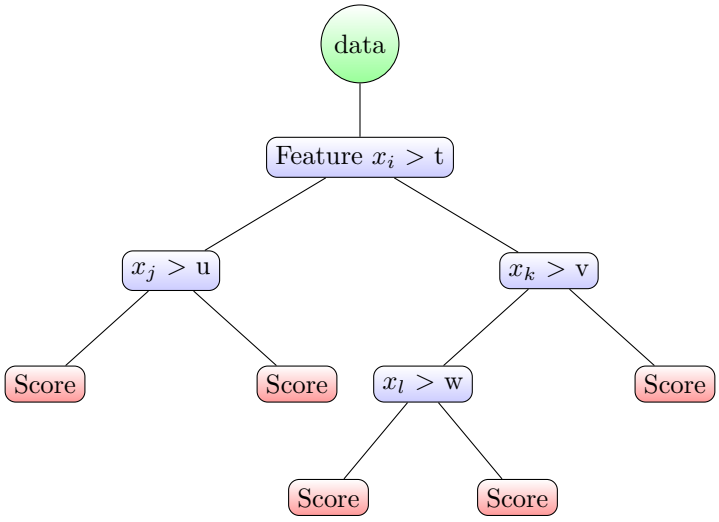

Figure 1: Binary tree structure

predictions on samples of unknown nature, e.g. the real data. The training data is split into two samples: one for the training (aka the fitting) and one for testing the result. The tree fitting is done using the following procedure:

1. Initialise: each data point gets an initial prediction (often a mean value, but can be random)
2. Calculate difference between predicted and actual value
3. Fitting: Fit the tree, e.g. minimise the difference between actual and predicted value
4. update predictions: Update the old predictions for each particle to the value after going through the tree
5. Repeat Repeat steps 2-4 for a specified number of trees

All these trees togther are then called a binary classifier.

2.2 Overfitting and hyperparameter

There are different hyperparameters to define a binary classifier, for example the maximal tree depth, the number of trees etc. The right choice of hyperparameter is important, because they prevent to overfit. One can imagine, that if the number of nodes is of the same magnitude as the number of data points used for fitting, the classifier will classify by the actual classification and not by the features. This means, that it is not able to make predictions for other data samples and is therefore useless.

2.3 Multiclass classification

There are two possibilities to do a multiclass classification with *xgboost*: The „one vs. all“method and the „one vs. one“method. In the first one, for n different classes, n binary classifiers are needed,

one for each class. Every classifier will calculate a score for the point to belong to this class or not. A final classification is done by choosing the classifier with the highest score.
The second method needs $\frac{n(n-1)}{2}$ binary classifiers to distinguish between n classes. Here every classifier distinguishes between a pair of classes and returns a score for this binary classification. The final score for each class can be calculated by combining all scores of the pairwise classifiers.
In summary, at the end of a multiclass classification, n BDT-scores are returned for every data point, one score for each class.
In this project the „one vs. one“ method was used.

3 Data preparation

The data we study comes from pp collisions at a centre-of-mass energy of 13.6 TeV collected by ALICE during the Run 3 data. To train the machine learning tool to classify prompt and non-promptΛs and background, artificial generated Monte Carlo data, corresponding to this process, is used.
The generated data is labeled with the PDGCode as well as with the PDGCode of the mother particle, e.g. the particle that decayed into the respective one. To get the prompt from the Monte Carlo sample, one has to choose all data points with the PDGCode of the (anti-)Λ and a mother PDGCode equal to 0, that means it comes from the original collision. The non-prompt generated particles are categorised by the PDGCode of the (anti-)Λ and a mother PDGCode that is non-zero. For each of this data point we have the following features available to train the machine learning algorithm:

- DcaV0Tracks $\rightarrow$ minimal distance of the decay products of a V0 particle
- DcaV0PV $\rightarrow$ minimal distance of the V0 to the primary vertex (the point, where teh collision took place)
- DcaPosPV $\rightarrow$ minimal distance of the positive charged particles to the primary vertex
- DcaNegPV $\rightarrow$ minimal distance of the negative charged particles to the primary vertex
- Ct $\rightarrow$ Decay length in the Lambda rest frame
- CosPA $\rightarrow$ the cosine of the angle between the V0 and the decay products

We therefore know how the distributions for these properties look like for prompt or non-prompt particles and they can be used during the learning process. In addition we know the invariant mass of each candidate.

3.1 Signal and background in real data

The background contribution in the real data is hard to predict. If it is very small one can perform a binary machine learning classification between prompt and nonprompt. To estimate the background contribution in the data, the invariant mass distribution can be fitted with a Crystal-ball+background function,

$$f(m) = \mathrm{CB}(m) + \underbrace{T(m)}_{\text{2. Order Poly}} \tag{1}$$

where the gaussian part of the distribution is placed at the invariant mass of the lambda, e.g. at ≈ 1.1155 GeV. The background is chosen to be a chebychev polynomial of second order, because it converges reliably using *RooFit*, but in principle one can choose any second order polynomial for the fit. The fit is shown in fig. 2.

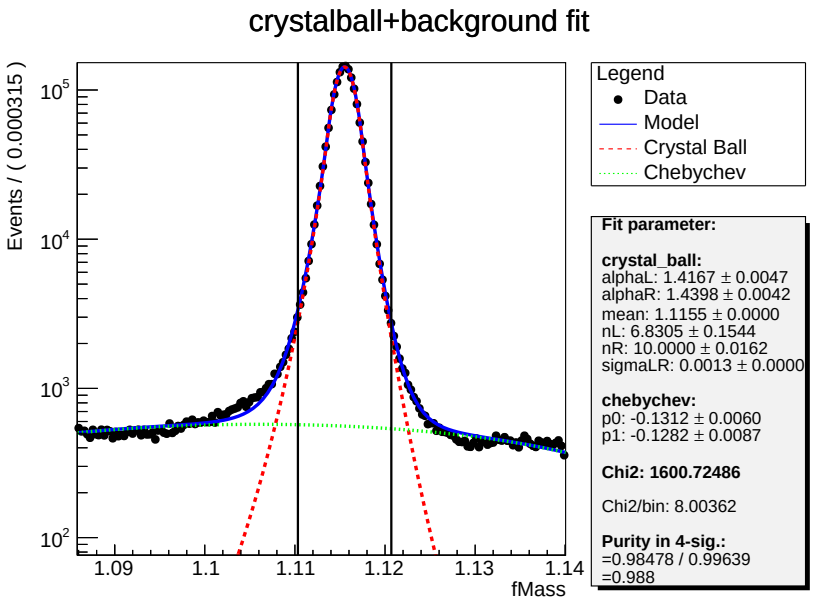

Figure 2: Mass distribution of data, fitted with a crystalball function + quadratic background

The background contribution can be expressed via the purity:

$$P = \frac{\text{signal (e.g. } \lambda\text{-particles)}}{\text{signal+background}} = \frac{\int_{\mu-4\sigma}^{\mu+4\sigma} \text{CB}(m)dm}{\int_{\mu-4\sigma}^{\mu+4\sigma} f(m)dm} \tag{2}$$

which should be close to one (small background contribution).

3.2 Background in the Monte Carlo sample

In addition, a background of the Monte Carlo data can be found by filter only the particles with PDGCode -999 (particles, that are not Λ-baryons). It is expected that the background has an equal mass distribution but in fact one observes a peak around the invariant mass of the Λ. A reason for this can be that muons, which are decay products of pions, which are in turn decay products of a Λ, are propagating approximately in the same direction as the pion (because they have approximately the same mass as the pion). Therefore, they are matched with a proton to a V0 instead of the pion mother, and therefore considered as background, but in reality they belong to a signal process and have the invariant mass of a λ. To check this assumption, we added the PDGCode of the Grandmother of the pion daughter (e.g. the mother of the pion), if it has the same Monte Carlo label as the mother of the proton to the MC table in the *O2Physics* code. The result is, that there

are indeed muons matched to the proton, and extracting them removes the bump in the invariant mass distribution as it can be seen in fig. 3. The distribution still has a bump around 1.14 GeV.

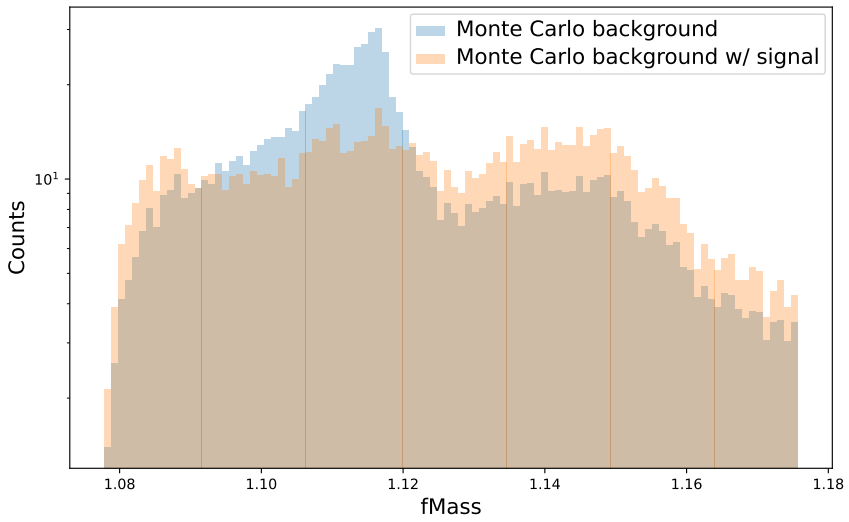

Figure 3: Mass distribution for the MC background with and without filter

The Armenteros-Podolanski plot fig. 4 shows a non trivial distribution of particles in the region of K0 shorts [3]. The K0 shorts can be removed by setting a Qt cutoff of 0.12. The resulting mass distribution in comparison to the previous one can be seen in fig. 5. The bump is clearly removed. We will referred to this prepared data set as "bckg MC wo Signal" in all following plots. In the machine learning both background samples (from data and from Monte Carlo) can be used and the output should be compared and analysed to understand the sensitivities of the machine learning algorithm.

3.3 Adding features

The Λ baryon decays weakly into a proton and a Π^- and respectively the anti-Λ decays into a anti-proton and a π^+. The particles are labeled with a positive Pt (transverse Momentum), the antiparticles with a negative Pt. Moreover, we know that positive particles will have a positive curvature inside the detector and negative particles will have a negative curvature. This means that we can add to the DcaPosPV and DcaNegPv feature (the features that describe the minimal distance between the tracks of the positive/negative decay products to the primary vertex) the DcaProtonPV and DcaPionPV that describe the minimal distance between the tracks of the (anti-)Protons /(anti-)Pions to the primary vertex (all data points with positive Pt and DcaPosPV or negative Pt and DcaNegPV are protons and anti-protons and the other way around for the pions). These features will be used in the machine learning process. The expectation is, that they improve the learning process and have a significant impact for the classification.
In fig. 6 one can see the distributions of the features used for the machine learning for the prepared prompt, nonprompt and Monte Carlo background sample. We see a strong difference in the distributions for the DcaV0Tracks between the background and the signal sets, so one expect this

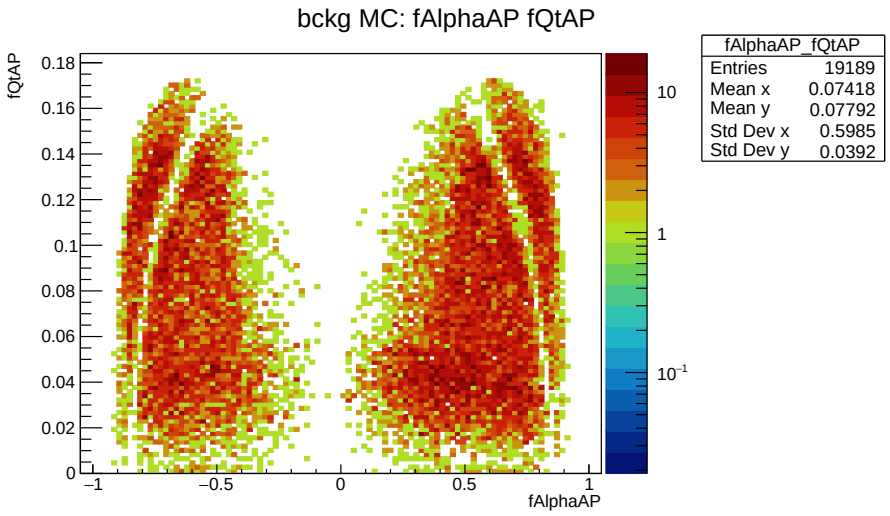

Figure 4: Armenteros-Podolanski plot for the Monte Carlo background

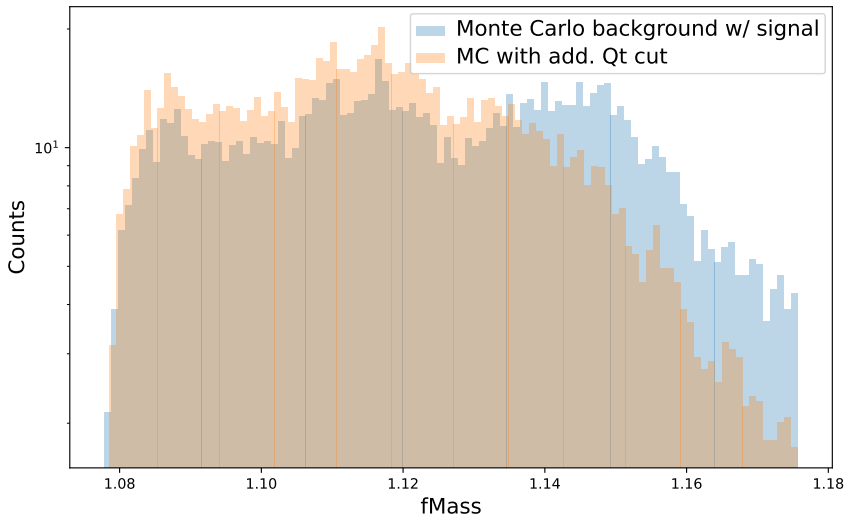

Figure 5: Mass distribution of the Monte Carlo background after the additional Qt cut

feature to be dominant for the classification of the background. Moreover, the features CosPA and DcaV0PV (which are strongly correlated) are looking clearly different for prompt and nonprompt, such that they should be the most important to distinguish these two classes.

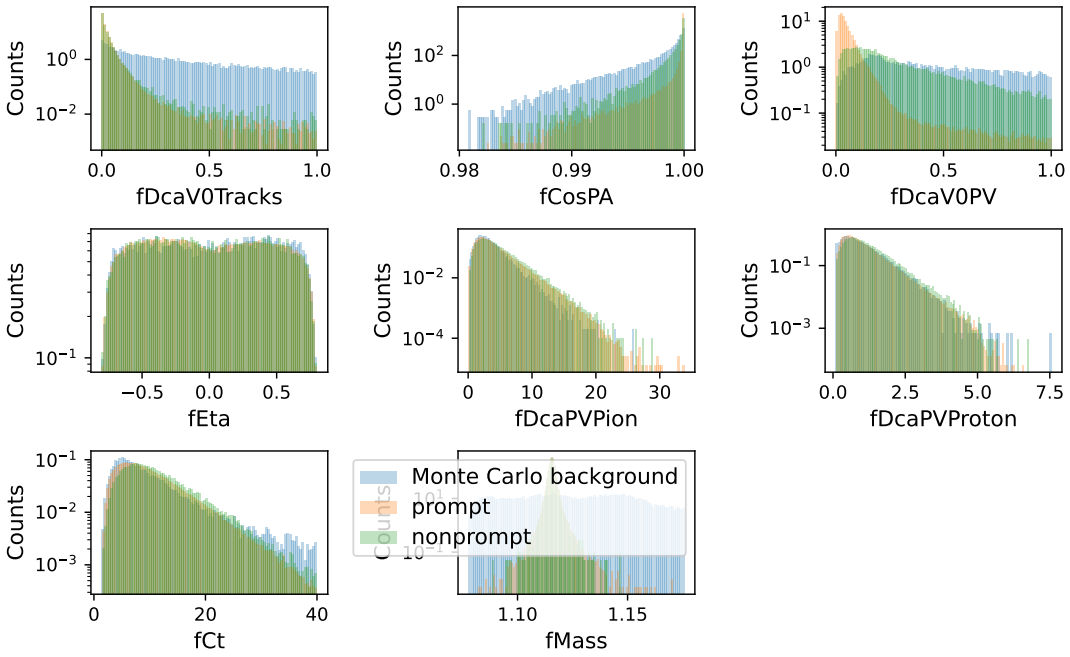

Figure 6: Feature distributions

4 Classification of Λ signals with Machine Learning

In the specific case of classifying Λs, the machine learning can solve a classification task using a machine learning model that classifies reconstructed lambda candidates as either prompt, non-prompt, or background.

4.1 Training and testing the model

To train the model we use the prepared data from section 3 for which we know the class for each candidate. The *hip4ml* module can check the feature importance [1]. The feature importance is related to the impact one feature has on the final BDT output. The most important features are generally the most discriminative among the analysed classes. The resulting feature importance of the training can be seen in fig. 7. The score on the x-axis can be interpreted as a quantification of the impact on the model output, color-coded for each class; as higher the score as more important the feature for the classification. One can see, that the most important feature for dividing prompt

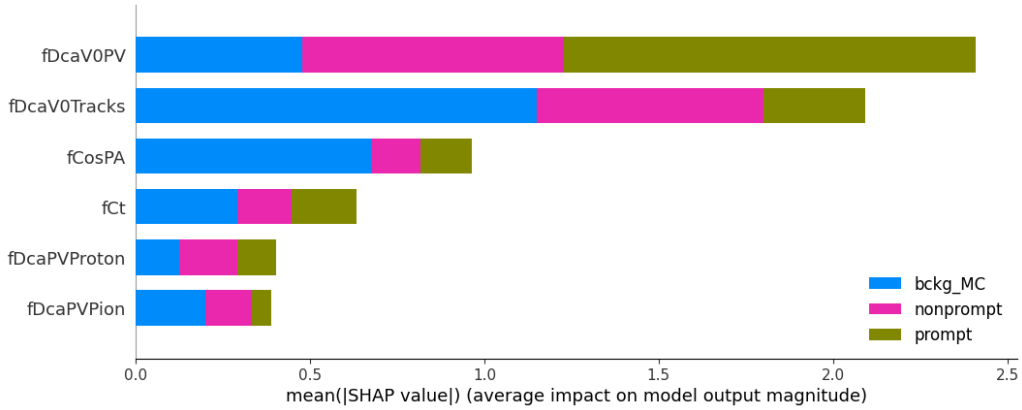

Figure 7: Feature importance for the training with Monte Carlo samples

and non-prompt Λs is the DcaV0PV, which is strongly correlated to the CosPA. The correlation appears, because as larger the pointing angle of the V0 (and therefore as smaller the cosine), as larger the smallest distance between the V0 and thr primary vertex. This is visualised in the sketch in fig. 8. The correlation can be pointed out by repeating the training without the DcaV0PV. The result of the feature importance for this can be seen in fig. 9. In this test, the CosPA becomes the most important feature, as it was to be expected. Moreover the most important feature to subtract the background is the DcaV0tracks. This is expected because the V0 tracks coming from the signal should come very close at the V0 vertex, but for teh background this is not the case.
To test a model, only one half of the prepared Monte Carlo data is used for the training and the other half can be plugged in after the training, to compare the output distributions for both the training and the testing data set. This shows if the model is able to generalize its performance to a statistically independent data sample. To quantify the goodness of distinction, one can have a look at the ROC-Plot fig. 10. Here, the trade-off between the true positive rate (particles that are correctly classified to the mentioned class) and the false positive rate (particles of another class that

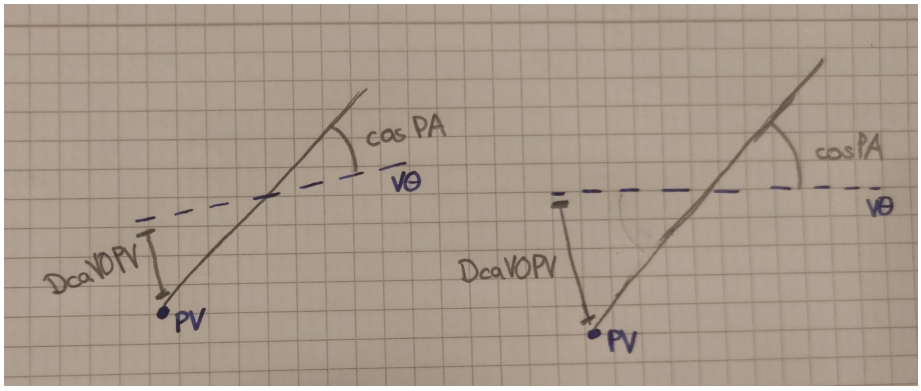

Figure 8: Visualisaztion of the correlation between the cosPA and the DcaV0PV

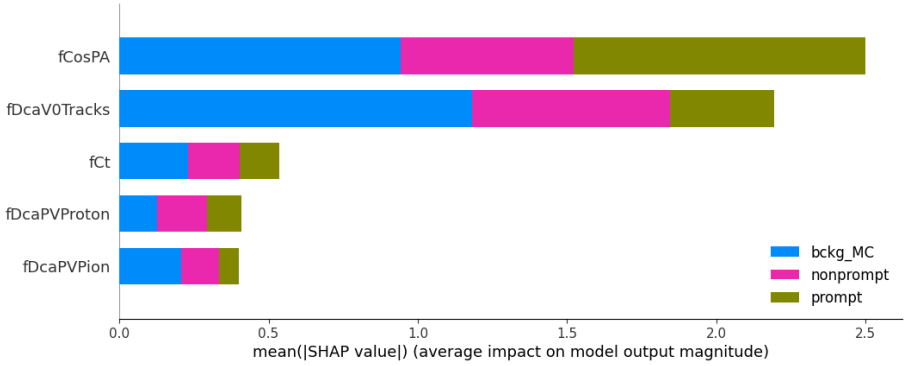

Figure 9: Feature importance for training without DcaV0PV

are wrongly classified to the mentioned class) at different thresholds is shown. The ROC-curves are calculated in One can see that the curves for the comparison of prompt against background are less equal to the „luck “ straight line, which means that the trained model can distinguish them very well and that the compromise between false positive classification and true positive classification is small (e.g. if one accepts a rate of 0.1 false positive classified prompt candidates, one receives a true positive rate of $\approx 95\%$). The trade-off between the true positive rate of non-prompt and the false positive rate of background is the most serious.

4.2 Characteristic BDT output score distributions

The distributions for the BDT output scores are visualised in a BDT output diagram, as shown in fig. 11 for class 2. This output score is connected to the probability of the candidate to be a prompt Λ. As it can be seen, the distributions for the training and the test sets look comparable, which indicates, that one avoided to overfit. The output score distribution of prompt candidates of the training set as well as the test set has a high peak close to one, which indicates that the prompt signal can be classified very well. The same type of diagram can be plotted for the BDT output

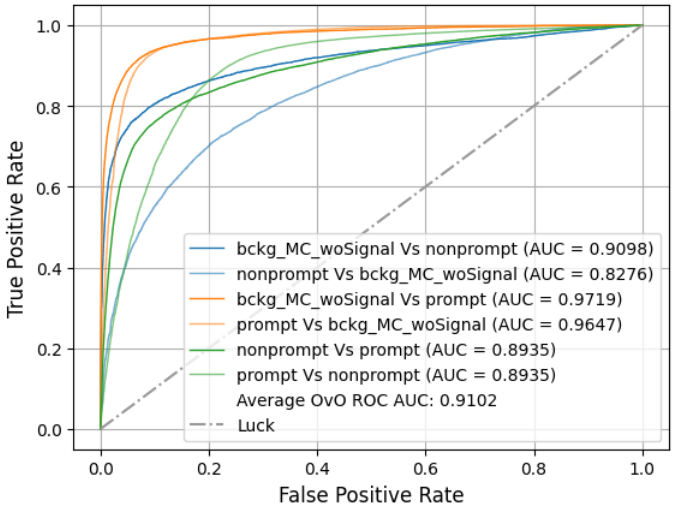

Figure 10: ROC plot for looking one vs one pairwise at the classes: background, non-prompt and prompt

score for class 0 and 1.
The distributions for the BDT scores of the Monte Carlo samples are characteristic for the model, which means the model should return a distribution with the same shape as the green one in fig. 11 for a set of data, containing only prompt particles but normalised to the number of data points.

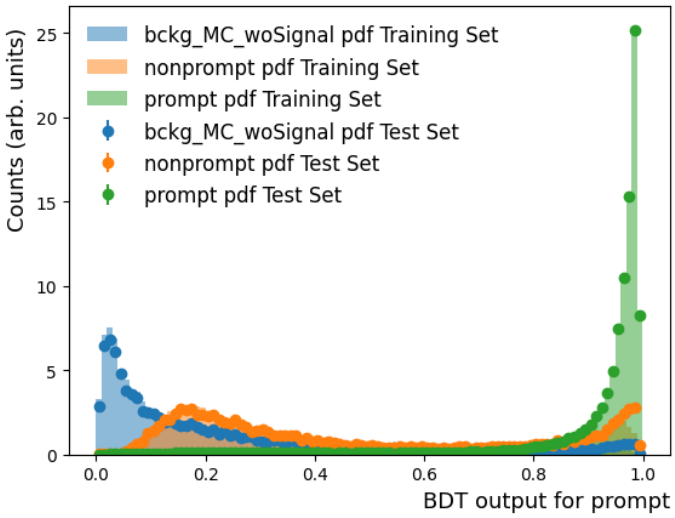

Figure 11: BDT output for test and training set

5 Results

The goal of the applied machine learning is to classify signal candidates and to get the contribution of prompt/nonprompt Λs in the signal data. Therefore we first apply the trained model to the data and receive the BDT output score distributions for each class. Now, we can perform a simultaneous fit, using *RooFit*, that fits the background plus signal to the 1) the BDT output histogram of class 0 (background) and 2) the BDT output histogram of class 2 (prompt). The fit functions for the BDT output are the mentioned characteristic BDT distributions of the model, each scaled with a factor (the fraction of the respective class with respect to the size of the whole data sample), that has to be fitted. The result can be seen in fig. 12. One can see that the method works well, as

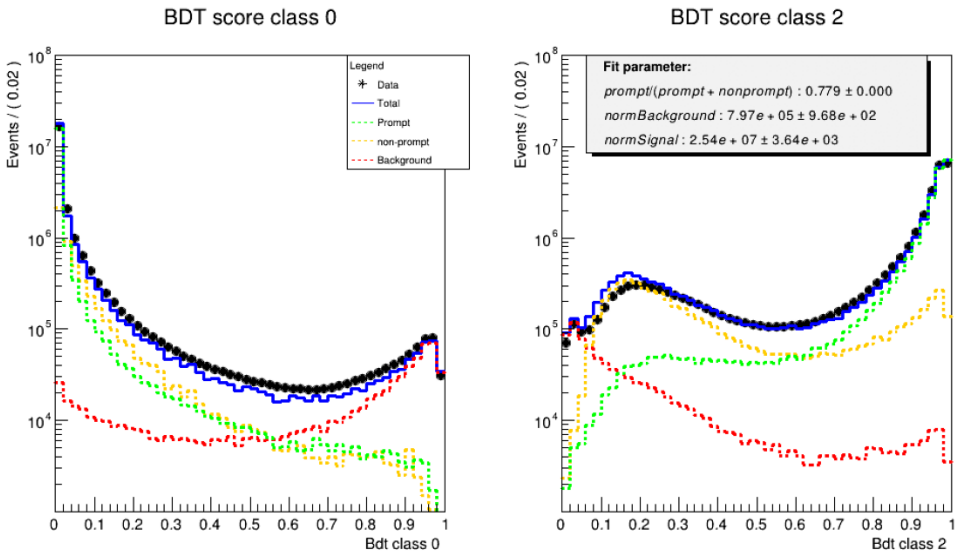

Figure 12: Result for simultaneous fit of the bdt output for class 2 (prompt) and class 0 (background)

both diagrams are fitted confidentially, but there is a discrepancy. Reasons for this could be that the MC might not be reproducing accurately the features observed in the data, hence the machine learning algorithm response on Monte Carlo differs from the one in the data and therefore the BDT output shape expected in MC could not match the one observed in data. Moreover, there might be some contamination effects in the data that are not considered in the Monte Carlo templates, for example, we have seen that the peak in the mass distribution in the Monte Carlo background around the lambda mass was not mentioned. The different contributions of prompt (green), non-prompt (orange) and background (red) are plotted individually. The result is a fraction of 79.9% of prompt particle in the Signal contribution. The result for the prompt fraction can be tested by doing the simultaneous fit for the Monte Carlo test set and compare it to the actual fraction. This check can be seen in the table fig. 13. The check/ closure test is required to validate the method, because the comparison of the method to a known or expected result, ensures consistency and identifies any discrepancies. One can see that the prompt ratio of the Signal resulting from the fit is close to the actual value.

Prompt/(Nonprompt+prompt), test data set

sim. Fit class 0 vs 2	MC truth
0.874 ± 0.001	0.876

Figure 13: Prompt fractions resulting from simultaneous fits vs actual ratio

References

[1] Luca Barioglio et al. *hipe4ml/hipe4ml*. Version v0.0.14. Aug. 2022. DOI: 10.5281/zenodo.7014886. URL: https://doi.org/10.5281/zenodo.7014886.

[2] 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*. KDD '16. ACM, Aug. 2016. DOI: 10.1145/2939672.2939785. URL: http://dx.doi.org/10.1145/2939672.2939785.

[3] J. Podolanski and R. Armenteros. "III. Analysis of V-events". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 45.360 (1954), pp. 13–30. DOI: 10.1080/14786440108520416. eprint: https://doi.org/10.1080/14786440108520416. URL: https://doi.org/10.1080/14786440108520416.