CERN Summer Student Program

Summer project:

BDT application to dark photon searches in H->y+yd with the ATLAS detector at the LHC

Sanad Binhalabi

United Arab Emirates University

Supervisors: Hassnae El Jarrari (CERN), Rachid Mazini (The University of Witwatersrand)

August 2024

Abstract

This study investigates the application of a Boosted Decision Tree (BDT) model in the classification of signal events and background noise, specifically within the context of exploring dark photon phenomena. The model was optimized using hyperparameter tuning techniques, including adjustments to parameters such as `num_leaves`, `feature_fraction`, and `max_depth`, aimed at enhancing predictive accuracy and model performance. A thorough analysis of feature importance was performed to identify key drivers that significantly impact the model's classification capabilities. The findings demonstrate the BDT model's potential in identifying and classifying events related to dark photons, with the results providing valuable insights into the model's effectiveness in distinguishing signal events from background noise. Future efforts will focus on further refining the model and expanding the dataset to improve its classification capabilities in the ongoing investigation of dark photons.

1 Introduction

The **Standard Model** of particle physics serves as the cornerstone of our understanding of the universe, describing the known fundamental particles and forces except gravity. However, it falls short in explaining phenomena such as dark matter and dark energy [1]. To explore these mysteries, many theoretical models propose the **Dark Photon**, a hypothetical particle that could bridge the gap between the visible and dark sectors of the universe, potentially mediating interactions involving dark matter [2]. These investigations are conducted using the **Large Hadron Collider (LHC)**, the world's most powerful particle accelerator, located at CERN. The LHC enables collisions at unprecedented energies, providing a critical platform for probing the Standard Model and searching for new physics [3]. Within the LHC, the **ATLAS** experiment plays a pivotal role in detecting and analyzing the byproducts of these high-energy collisions. By analysing the data collected from ATLAS, scientists aim to uncover the fundamental properties of matter and explore the potential existence of new particles such as the Dark Photon [4]. In our case, we use BDT as classification in the analysis of the data we have. One signal and six background data samples were used in this analysis to train and optimize the hyperparameter space in order to obtain the best classification and discrimination.

2 Machine Learning approach in HEP

In high energy physics (HEP), machine learning (ML) plays a crucial role in managing and interpreting the vast amounts of data generated by experiments at particle accelerators like CERN's Large Hadron Collider (LHC). ML algorithms help physicists filter signal events from background and noise ones, classify particles, and identify rare events among billions of collisions, improving both the speed and accuracy of data analysis. Techniques like deep learning boosted decision trees, and neural networks are widely used for tasks such as particle identification, event reconstruction, and anomaly detection. This integration of ML in HEP not only enhances our ability to test theoretical models and discover new particles but also pushes the boundaries of both fields by solving complex problems through collaborative innovation. In our project, we have used the BDT method in the classification process to recognize the signal from the background.

2.1 BDT in HEP

In our project I used a boosted decision tree (BDT) method. A boosted decision tree is a machine-learning model that improves the performance of decision trees through an ensemble technique called boosting. It sequentially trains multiple decision trees, with each tree focusing on the errors of the previous one. After each tree is added, the training data is reweighted to emphasize misclassified instances, compelling subsequent trees to concentrate on these difficult cases. The final prediction combines the weighted outcomes of all trees, enhancing accuracy and reducing overfitting. Common algorithms include AdaBoost and Gradient Boosting, which are widely used

in complex classification tasks across various industries. a very known method in using classification in machine learning. We applied it in our data where we had been dealing with a signal file and six background files. The process we make is to investigate the presence of dark photons, by analyzing the data and dealing with the missing energy in the transverse plane.

2.1.1 data

A signal file and six background files were our data. where the background was the majority of our data. all were ROOT files. To deal with such data we learn how to manipulate them using Python and using famous dataframes like pandas. First, we took the tree of the signal file and created a pandas data frame so we could see all the events and their data. Then we did the same with the six backgrounds. The signal file contains about 33,847 events, the all backgrounds file contains 6,318,966 events and the total when we concatenate all of them was 13,201,445 events. and the huge number of events and the size of the data made the process very slow. but because we use cuts to maximize the best result, the data has been deducted into a small number of events relatively, where the number of events after the cuts was only 954,253 events.

2.2 features plotting

In **Fig[1]** you can see the features have been plotted, and each one on the plotting table contains one feature. Features are: *'met_tst_sig'*: The significance of the total scalar sum of transverse momenta (missing transverse energy), quantifying how statistically significant it is *'met_jetterm_et'*: The transverse energy component of the missing energy vector associated with jets., *'met_jetterm_sumet'*:The scalar sum of transverse energies for all objects contributing to the jet term, *'met_tst_phyterm_phi'*: The azimuthal angle of the missing transverse energy vector calculated using the physics object term , *'abs_ph_eta':The absolute value of the pseudorapidity (η\etaη) of the photon(s) involved.*, *'jet_cerntral_pt_0':The transverse momentum of the leading central jet.*, *'new_met_tst_noJVT_et':The transverse energy of the missing transverse momentum vector calculated without applying the Jet Vertex Tagger (JVT).*, *'delta_jet_central_phi':The difference in azimuthal angle between the central jets.*,*'met_tst_jetterm_phi':The azimuthal angleof the missing transverse energy vector calculated using the jet term.*,*'new_jet_centerla_eta':The pseudorapidity of the new central jet(s).*,*'abs_jet_central_pt':The absolute value of the transverse momentum of the central jet(s).*,*'met_jetterm_phterm_phi':The azimuthal angle difference between the jet term and the photon term in the missing transverse energy calculation.*,*'abs_jet_central_timing':The absolute value of the timing information of the central jet(s), typically used to assess the jet's interaction time.*, *'failJVT_jet_pt_0': The transverse momentum of the leading jet that fails the Jet Vertex Tagger (JVT) requirement.*

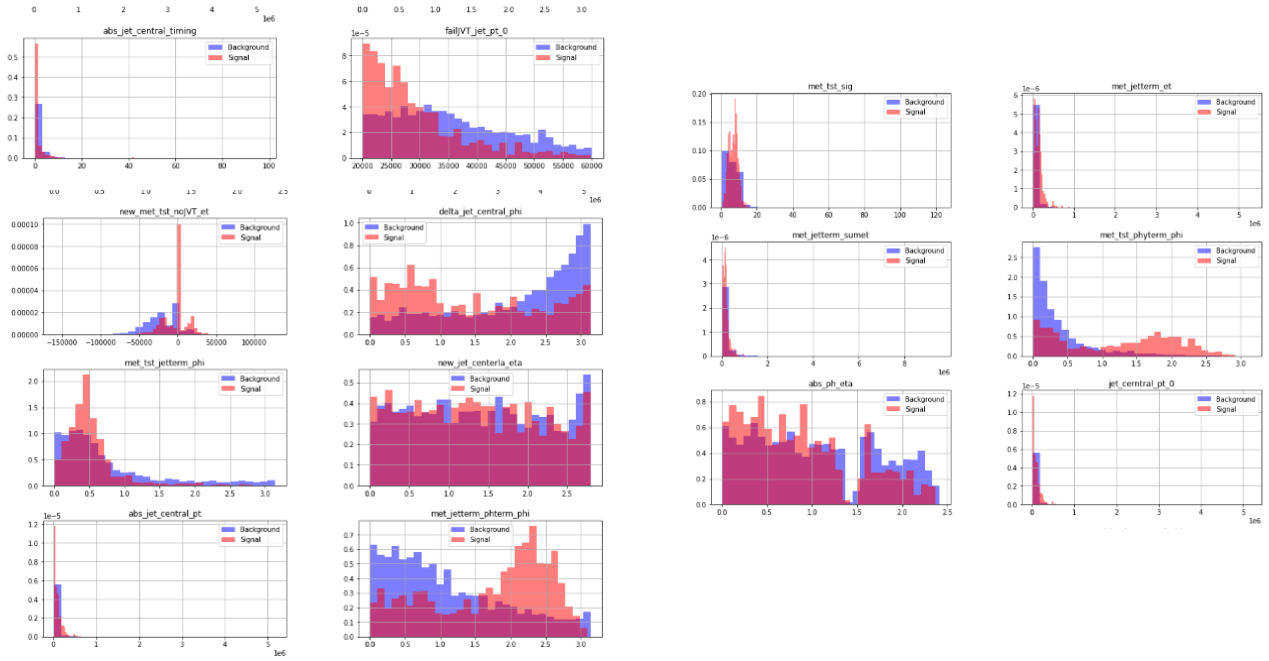

Figure 1: plotting the features.

2.2.1 Correlation Matrix

A correlation matrix is a table that shows the correlation coefficients between variables in a dataset, helping to identify relationships between them. Values range from -1 to +1, indicating the strength and direction of linear relationships. In **Fig[2]** correlation matrices have been plotted for both a signal and Background. After plotting the data we need to see the features' importance and then come back to the correlation matrix and exclude the least important features that has a very strong relation with other features. this should be done because it helps to reduce redundancy and multicollinearity. This streamlining process prevents the model from overfitting to noise, improves interpretability, and enhances computational efficiency by focusing on the most influential features. As a result, the model becomes more robust and better generalizes to new data.

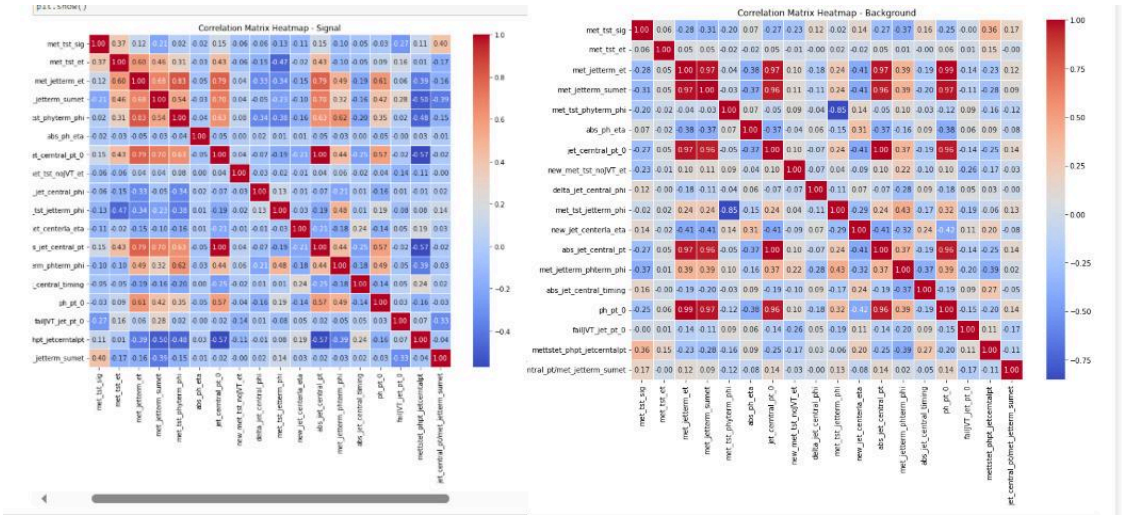

Figure 2: correlation matrix, one represents the signal and the other represents the background.

2.2.3 features importance’

In Boosted Decision Tree (BDT) models, feature importance refers to the influence each feature has on the predictive performance of the model. Features with high importance contribute significantly to the model’s accuracy by providing essential information for decision-making. These features are typically identified through the model’s iterative learning process, where the BDT algorithm emphasizes features that reduce prediction error. Conversely, features with low importance have minimal impact on the model’s performance and might be redundant or irrelevant. By analyzing feature importance, we can gain insights into the underlying patterns within the data, identify key drivers of the target variable, and optimize the model by excluding less significant features. In **Fig[3]** , all features are plotted where the most important one of them is in the top and the least important is in the bottom.

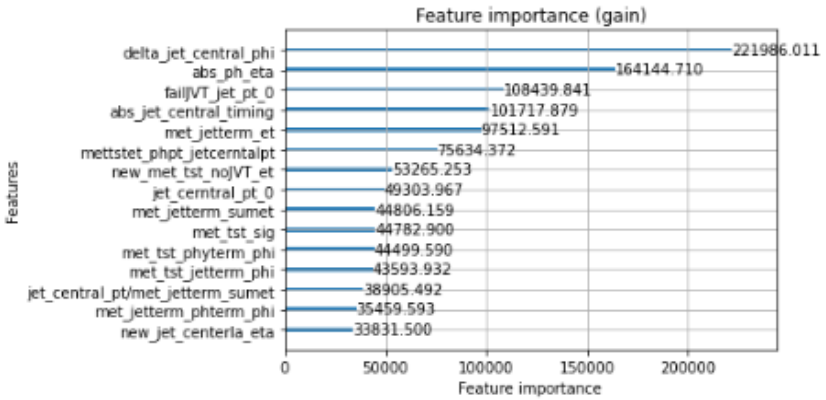

Figure 3: features important as delta_jet *centra_phi is the most important features*

3. Train the data and model

In the train part we used train_test_split function from the sklearn library. we split our data into : train and test x ,train and test y , and train and test weight. where x represents the features , y represents the labels (1 for signal and 0 for background), and weight is the calculated weight for each event.we set our test train size to be 0.75 of all events and the the test size is 0.25. then we used lightgbm as our model for BDT. in **Fig[4]** we can see the ROC of our data. The colors are represented by the number of data that has been use . The black color represents all data. I found the AUC (Area Under the Receiver Operating Characteristic Curve) of the test data to be 0.8443 and theAUC train to be 0.9976 , and if I exclude the weight then the AUC test would be 0.91895.An AUC score of 0.84429 suggests that the model performs well in distinguishing between the positive and negative classes. The AUC score ranges from 0 to 1, where a value of 0.5 represents random guessing and 1.0 signifies perfect classification. With an AUC of 0.84429, the model demonstrates a high level of accuracy in predicting the positive class over the negative class, reflecting strong discriminatory power and overall effectiveness in distinguishing between the two classes. This high AUC value indicates that the model is proficient at ranking positive cases higher than negative ones, which is crucial for applications where accurate classification is important.

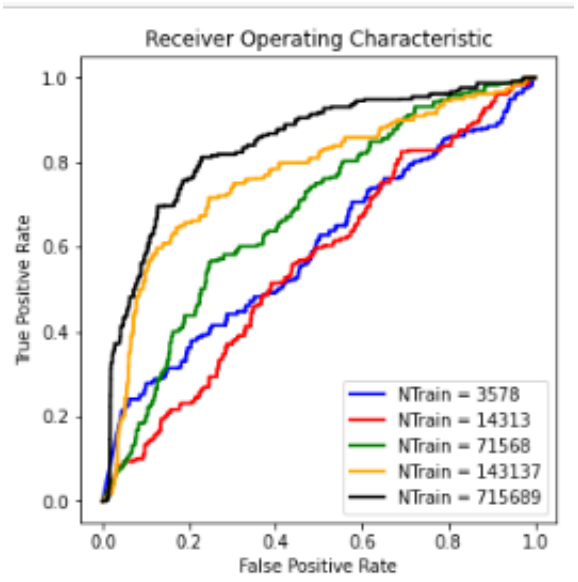

Figure 4: ROC for the data before dosing any hyperparameter optimization.

In **Fig[5]** the BDT significance is graphed where its score is 6.8025548968331595. This high value suggests that the model's performance is statistically significant, indicating that the observed results are unlikely to have occurred by chance. This level of significance supports the robustness and reliability of the GBM model in making accurate predictions or classifications for our dataset.

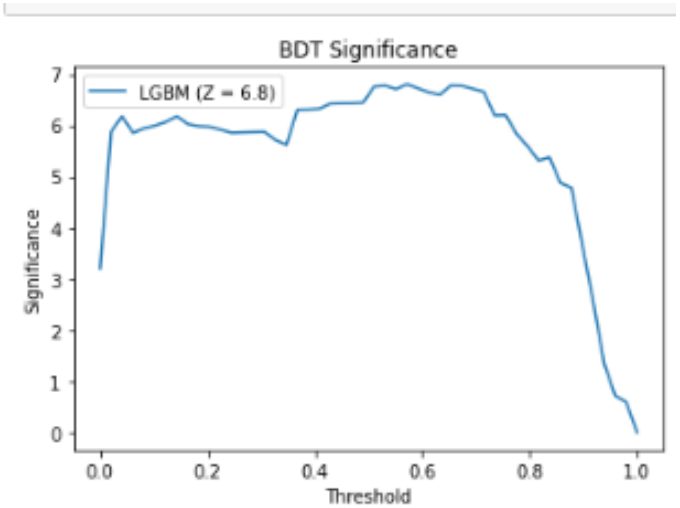

Figure 5: The graph shows the BDT significance. looks weird.

And **Fig[6]** shows the BDT score, where the red part represents the signal and blue colour represents background. This graph is important and shows if our model can distinguish between background events and signal events. The graph displaying the BDT scores reveals that the separation between signal and background is not as distinct as we would hope. The red columns, representing signal events, do not show a clear separation from the blue columns, which represent background events. This lack of clear differentiation suggests that the model may be struggling to effectively distinguish between signal and background. The overlapping of red and blue columns

indicates that the BDT might not be achieving the desired classification accuracy, and further tuning or additional features might be needed to improve the model's performance. This insight highlights the need for potential adjustments in the model or additional analysis to enhance its discriminative power and overall effectiveness. Thus we need to find a better why to make our model recognize the signal and background.we will try to find the best hyperparameters to enhance our model.

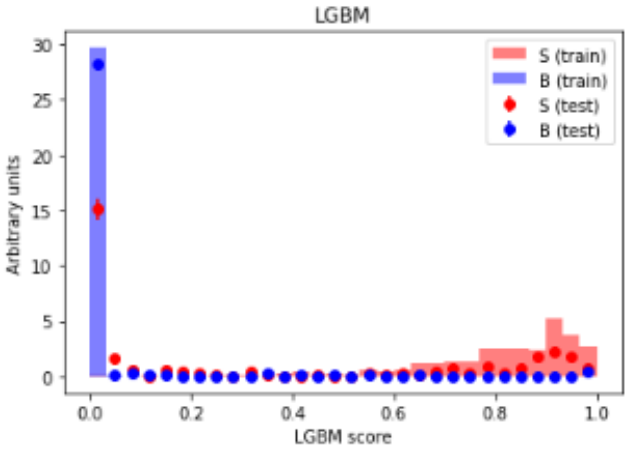

Figure 6: BDT score

3.1 Hyperparameters optimization

Hyperparameter optimization in classification with Boosted Decision Trees (BDTs) involves systematically tuning parameters to improve model performance. Key hyperparameters include the number of trees, learning rate, tree depth, and the minimum samples per leaf. Techniques like grid search, random search, or more advanced methods such as Bayesian optimization are used to find the optimal combination of these parameters. Proper optimization helps enhance the model's accuracy, prevent overfitting, and ensure it generalizes well to unseen data. in our model, used the following hyperparameters: The hyperparameters I used for optimizing the Boosted Decision Tree (BDT) model include `num_leaves`, which controls the number of leaves in each tree to balance complexity and generalization, with values like 24 and 45. `Feature_fraction` determines the proportion of features used for building each tree, with smaller values (0.1) promoting diversity and larger values (0.9) potentially improving performance if underfitting. `Bagging_fraction` specifies the fraction of data used for training each tree, where 0.5 introduces diversity to reduce overfitting and 1 uses the full dataset, which can increase overfitting risks. `Max_depth` restricts tree depth to control overfitting, with limits such as 5 for simplicity and 8.99 for capturing more complex patterns. `Lambda_l1` and `lambda_l2` apply L1 and L2 regularization respectively, with values like 0 and 5 for L1 and 0 and 3 for L2 to manage feature selection and prevent overfitting. `Min_split_gain` defines the minimum gain needed to make a split, with higher values like 0.1 ensuring only significant splits are made, while `min_child_weight` sets the minimum sum of instance weight required for a child node to further split, where lower values (5) allow more splits and higher values (60) promote conservative growth. Tuning these parameters helps balance model complexity and enhance performance on unseen data.

4 Results

After applying the hyperparameter to our model, the **Fig[7]** shows the BDT score for the result. And in **Fig[8**] we can see the ROC graph. The AUC test we found is about 0.797. Unfortunately the updated hyperparameters, intended to improve the model's

accuracy and generalization, have led to a less distinct separation between signal and background in the graph. This indicates that the model's ability to differentiate between the two classes has worsened. Moreover, the performance metrics show a decline, suggesting that the optimizations may have introduced issues such as overfitting or underfitting. This unexpected outcome highlights the need for further adjustment and careful reassessment of the hyperparameter choices to restore and enhance the model's effectiveness.

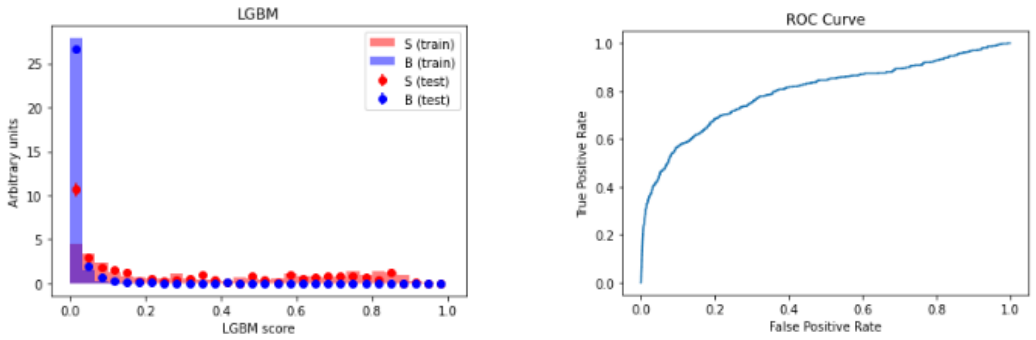

Figure 7: BDT score after applying HPO

5 Conclusion

The suboptimal results of our Boosted Decision Tree (BDT) model can primarily be attributed to the limited time and resources we had during this project. These constraints significantly hindered our ability to conduct a comprehensive hyperparameter search and fine-tune the model effectively. Consequently, the BDT score graph displays a poor separation between the signal (red columns) and the background (blue columns), indicating that the model struggled to classify the data accurately. Additionally, due to time constraints, we did not thoroughly evaluate feature importance or identify and exclude features with the strongest interdependencies. This lack of a robust feature selection process could have contributed to the model's inability to generalize effectively. The scarcity of data resources further limited the model's ability to learn and generalize, leading to issues such as overfitting or underfitting. This inadequacy in the data size and diversity compromised the model's capacity to handle the complexity of the classification task, resulting in a decline in performance metrics and overall accuracy. If we had more time, we could have conducted a more detailed analysis to identify the root causes of these issues, explored a wider range of hyperparameter configurations, and validated the model against a more diverse dataset. These additional efforts would have enabled us to refine our approach, optimize the model's performance, and achieve better and more reliable results. With extended time and enhanced resources, we are confident that we could have overcome these challenges and significantly improved the model's effectiveness in distinguishing between signal and background events.

References:

1. CERN: The Standard Model - https://home.cern/science/physics/standard-model
2. Nature Physics: Dark Photons - https://www.nature.com/articles/nphys3605
3. CERN: The Large Hadron Collider - https://home.cern/science/accelerators/large-hadron-collider
4. ATLAS Experiment Official Website - https://atlas.cern/