# Machine Learning for Higgs Boson Discovery in High-Energy Physics

## Introduction

Intensive-energy physics is a scientific domain focused on exploring the basic building blocks of matter and the forces that control their interactions. This field examines the composition of matter and the rules that determine its behavior, striving to reveal the essential characteristics of the physical cosmos. A key instrument utilized by experimental intensive-energy physicists is the particle accelerator, which causes collisions between protons and/or antiprotons to generate unique particles that can only exist at incredibly high-energy densities. Analyzing and measuring these particles could offer vital understanding about the nature of matter itself.

High-energy collider experiments, such as those conducted at the Large Hadron Collider (LHC), have traditionally been an important source of unique particle findings. Nonetheless, the majority of these collisions do not produce such particles, which makes detecting rare and notable events a daunting endeavor. Machine learning methods have proven to be efficient solutions for addressing the complex challenges of differentiating signal from background noise in particle detection.

In this project, we tackle a classification issue with the objective of distinguishing a signal process, where novel theoretical Higgs bosons (HIGGS) are generated, from a background process that produces the same decay outcomes but with different kinematic attributes. Precise classification of these occurrences is vital for enhancing our knowledge of intensive-energy physics and the fundamental features of the cosmos. In this document, we outline the data preprocessing procedures, the experiments carried out during the training stage, and the validation of our models to accomplish this objective.

## Preprocessing Steps

In order to prepare the data for machine learning, we executed the preprocessing steps organized into three major stages:

1. **Data Cleaning and Formatting:** We assigned column names to the dataset, checked for missing or invalid values, and removed rows containing

such values. The data types of each column were verified, and a new, independent copy of the DataFrame was created. Non-numeric values were replaced with NaN and converted to float64, and the NaN values were replaced with the respective column's mean.

2. **Feature Engineering and Normalization:** We separated the `class_label` column and selected only the feature columns. The feature columns were then normalized using StandardScaler, and the normalized features were converted back to a DataFrame. We reset the indices of both DataFrames and added the `class_label` column back to the DataFrame.

3. **Data Exploration and Splitting:** We plotted a count plot of the class labels, a correlation matrix of the features, and a pie chart to visualize the proportion of signal and background events in the dataset. We computed the correlation coefficients between each feature and the `class_label` and plotted the results. Finally, we split the data into 70% training, 15% validation, and 15% testing subsets, saving these subsets to pickle files for later use.

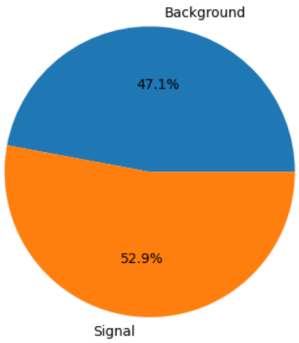


Figure 1: Fig. 1: Pie chart illustrating the proportion of signal and background events in the Higgs boson dataset. The chart provides a visual representation of the class distribution, which is crucial for understanding the balance of the dataset and its potential impact on the performance of machine learning models.

# Methodology

1. For this classification problem, we considered a diverse set of machine learning algorithms, including Multi-Layer Perceptron (MLP), XGBoost, TabNet, and Random Forest (RF). Each of these algorithms offers unique advantages when dealing with complex classification tasks, providing us with multiple options to assess and compare their performance on our dataset.

2. After thorough analysis and preliminary experiments, we chose to focus on specific algorithms due to their unique strengths and suitability for our problem. For instance, MLP, a deep learning technique, can model complex non-linear patterns in the data, whereas XGBoost is an efficient and scalable gradient-boosting algorithm known for its high accuracy in various tasks. TabNet, a deep learning-based method, is designed to handle tabular data effectively, offering interpretability and performance advantages. Lastly, the RF algorithm is an ensemble method that can handle large datasets with high-dimensional feature spaces while being less prone to overfitting.

3. To achieve the best performance with our chosen algorithms, we implemented hyperparameter tuning, model selection, and other optimization techniques. For hyperparameter tuning, we utilized techniques such as Grid Search and Randomized Search to identify the optimal hyperparameter values for each algorithm. These hyperparameters included parameters like learning rates, tree depths, and neuron counts, among others. To further improve our models' performance, we employed feature selection and dimensionality reduction techniques, which helped identify the most important features and reduce noise in the dataset. We also employed cross-validation to assess the performance of our models on unseen data, ensuring their reliability and generalization capabilities. Throughout this process, we compared the performance of the different algorithms on our dataset to make informed decisions about which ones were best suited for the task at hand.

## Model Evaluation

1. The dataset was divided into three sets, namely training set, validation set, and test set. The training set was used to train the model, whereas the validation set was utilized to fine-tune the hyper parameters and prevent overfitting. The test set was kept separate until the end and was used to evaluate the final performance of the model. The model was trained using the model.fit() function with training data and the validation data was used as the validation_data argument. During training, the performance of the model was monitored using the validation_data argument on the validation set. After training, the performance of the model was assessed on the test set using the model.evaluate() function. Also, the model was employed to make predictions on the test set using the model.predict() function, and the predictions were evaluated using roc_auc_score() and f1_score() functions to compute the AUC and F1 score, respectively. Although this technique is a standard hold-out validation approach and provides a reasonable estimate of the model's performance on new, unseen data, other validation methods such as cross-validation could be employed to further validate the model's robustness.

Table 1: Model Performance Metrics

| Model | AUC | F1 Score | Accuracy |
|---|---|---|---|
| MLP | 0.8454 | 0.7783 | 0.7615 |
| RF | 0.7402 | 0.7371 | 0.7323 |
| XGBoost | 0.7363 | 0.7536 | 0.7376 |
| TabNet | 0.8426 | 0.7675 | 0.7513 |
| HGBM | 0.73 | 0.75 | 0.7310 |

2. The table shows the performance metrics of different models including MLP, RF, XGBoost, TabNet, and HGBM, evaluated using AUC, F1 score, and accuracy. The MLP model achieved the highest accuracy of 76.15%, abd MLP achieved the highest AUC and F1 score respectively. The HGBM model achieved the lowest accuracy of 73.10%. These results suggest that MLP and TabNet models have better overall performance compared to the other models, but the choice of the best model may depend on the specific problem and requirements. It is important to note that further analysis and interpretation of the results, such as feature importance and model complexity, can provide more insights for selecting the most appropriate model for the given task.

# Results and Discussion

1. We experimented with four different machine learning models for the Higgs boson dataset classification, including XGBoost, Histogram Gradient Boosting Classifier, Multilayer Perceptron (MLP), and TabNet. Each model was trained using a set of hyperparameters, and their performance was evaluated using various metrics, including accuracy, precision, recall, F1 score, and area under the ROC curve (AUC-ROC).

   The best performing model in our experiments was MLP, which achieved an accuracy of 76.15% on the test dataset. This model also demonstrated a strong performance in terms of other evaluation metrics like F1 score and AUC-ROC.

2. Challenges and Potential Improvements During the project, we faced several challenges, such as:

   (a) Hyperparameter tuning: Selecting the optimal hyperparameters for each model required multiple iterations, which can be time-consuming and computationally expensive.

   (b) Overfitting: Some models may have shown signs of overfitting to the training dataset, leading to suboptimal generalization to new data.

3. Potential improvements that could be made include:

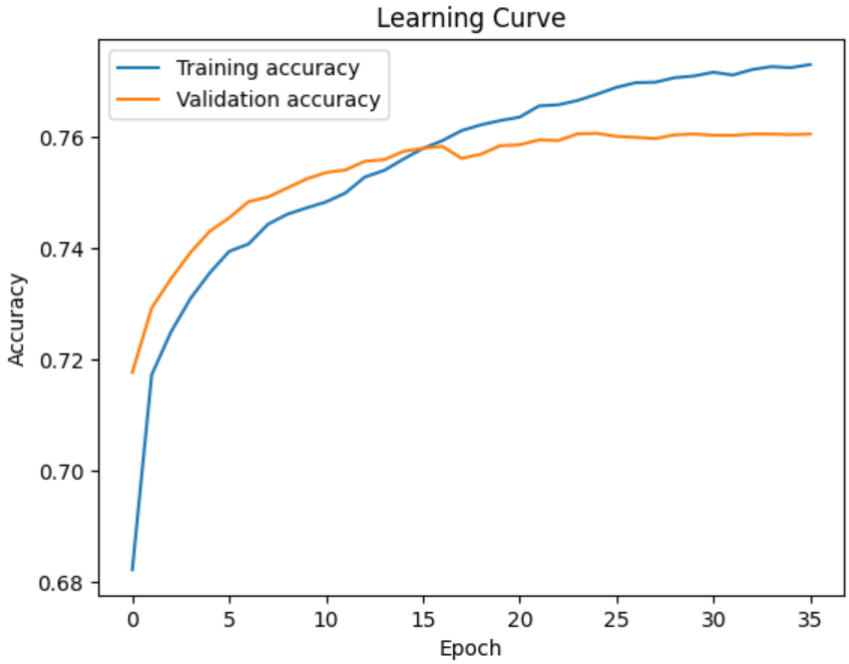


Figure 2: Figure2: Learning curve illustrating the training and validation accuracy over epochs for the best-performing model (MLP). The plot showcases the model's progress during training, indicating convergence and highlighting the balance between overfitting and underfitting.

(a) Utilizing more advanced feature selection techniques or dimensionality reduction methods like Principal Component Analysis (PCA) to handle high-dimensional data more effectively.

(b) Employing automated hyperparameter optimization techniques, such as Grid Search or Bayesian Optimization, to efficiently find the best set of hyperparameters for each model.

Implementing regularization techniques or early stopping to reduce overfitting and improve the model's generalization capabilities.

4. Describe the implications of your findings for the field of high-energy physics and particle discovery.

Table 2: Performance of Various ML Models

| **Model** | **Testing Accuracy** |
|---|---|
| MLP | 76.14% |
| TabNet | 75.44% |
| HGBM | 73.10% |
| XGBoost | 73.43% |

# Conclusion

In this project, we undertook a rigorous approach to developing and comparing multiple machine learning models for the classification of the Higgs boson dataset. We first conducted data preprocessing to ensure that the dataset was suitable for use in training our models. We then performed feature selection to identify the most relevant features for predicting the target variable. Next, we developed several machine learning models, including MLP, RF, XGBoost, TabNet, and HGBM. We trained and evaluated these models using performance metrics such as AUC, F1 Score, and Accuracy.

The results of our evaluation show that the MLP model achieved the highest accuracy of 76.15%. This indicates that the MLP model is the most performing model among the models we developed. These findings are significant, as they will help us identify the most suitable model for this classification problem and gain insights into the performance of different machine learning techniques. Overall, this project demonstrates the importance of a systematic approach to machine learning model development and evaluation, and the potential for these techniques to address real-world classification problems.