# Separating Higgs particle signals from background noise using Machine Learning

Computational Quantum Physics

Martha Papadopoulou

July 2024

## 1 Introduction

This project examines the signals of Higgs particles and investigates whether we can predict actual signals or background noise using machine learning. To achieve this, we use supervised learning with two classifiers, SVM and Random Forest, along with an artificial neural network, utilizing the tensorflow library. The objective is to assess the accuracy of each method for both low-level and high-level quantities.

## 2 Theoretical Framework

High-energy particle collisions are crucial for uncovering exotic particles. However, the difference between actual signals and background noise is difficult to distinguish and often necessitate the use of machine learning. The low-level quantities are basic measurements, such as the momentum of each observed particle, made by particle detectors. The data consists of twenty one feature sets. The high-level quantities, which consist of seven feature sets, are derived from low-level data to better capture the differences between signal and background. These features include the invariant mass of particle systems. By reconstructing these masses, high-level quantities provide greater discrimination power. (ref. [1])

## 3 Data Preparation

The data is divided into a signal column, a low-level matrix and a high-level matrix. These matrices are used separately to preform machine learning. To do so, the data from each matrix needs to be split into training and test sets, where the test sets comprise 20% of the total data size. The training and test data of both the low-level and high-level quantities are scaled, as it is required for SVM and ANN, however it is not needed for Random Forest. The following implementations of the methods are taken from the reference [2].

# 4 Supervised Learning

## 4.1 SVM Classifier

The SVM classifier maximizes the margin between classes for better generalization and offers flexibility with kernel methods to handle complex relationships. To employ this classifier, we utilize the parameters kernel = 'linear', random state = 42 and probability = True. The first parameter specifies that the kernel is linear, which implies that the decision boundary is also linear. We use the linear kernel for simplicity and computational efficiency. It serves as a good baseline to start with, allowing us to explore more complex kernels later if needed. The random state parameter sets a seed to ensure the reproducibility of results. The probability parameter enables obtaining probability estimates for predictions.

After defining the classifier, we use the scaled training data along with the signal data to train it. We predict the outcomes using our classifier and assess both the confusion matrix and the accuracy. The accuracy for the SVM classifier, using the low-level quantities is **0.5578** and the control matrix the following.

$$\begin{bmatrix} 328 & 460 \\ 248 & 565 \end{bmatrix}$$

Where the y-axis represents the true label and the x-axis the predicted label.

By applying the same process to the high-level quantities, we obtain the results for accuracy **0.6115** and the confusion matrix.

$$\begin{bmatrix} 272 & 516 \\ 106 & 707 \end{bmatrix}$$

## 4.2 Random Forest Classifier

Random Forest is a robust method known for its highly accurate predictions. It mitigates overfitting by averaging multiple decision trees trained on different subsets of the data. Additionally, it is robust to noisy data and provides estimates of feature importance, which aids in feature selection and interpretation. The first parameter n estimators = 50 is for the number of decision trees employed. The criterion parameter determines the function to measure the quality of a split, meaning the algorithm will use information gain as the criterion for splitting the nodes in the decision trees.

The Random Forest classifier does not require the data to be scaled, so we fit the original data to it directly. After predicting the values, we observe that the accuracy for low-level quantities is **0.5646**, and the confusion matrix is as follows.

$$\begin{bmatrix} 416 & 372 \\ 325 & 488 \end{bmatrix}$$

For the high-level quantities the accuracy is **0.6714** and the confusion matrix:

$$\begin{bmatrix} 521 & 267 \\ 259 & 554 \end{bmatrix}$$

# 5 Artificial Neural Network

Artificial Neural Networks (ANNs) are computational models inspired by the human brain's neural networks. They consist of interconnected nodes (neurons) organized in layers. Input data passes through these layers, where each neuron processes information.

To construct the ANN, we need to define its layers. The input layer has a shape equal to the number of columns in each dataset, which is 21 for the low-level quantities and 7 for the high-level quantities. The next layer has 32 neurons and activation relu, which is a function that outputs the input directly if it is positive, and zero otherwise. The following layer has 21 neurons and the same activation as the previous layer. The last layer returns one output with activation sigmoid, which is a function that squashes the output to a range between 0 and 1.

The next step is to compile the ANN, using optimizer adam, the loss function binary crossentropy and metrics accuracy. Optimizer adam dynamically adjusts the learning rate during training. The binary cross-entropy is a loss function used for binary classification and the accuracy is the metric used to evaluate the performance of the model. The scaled data is fitted to the ANN, with a batch size of 16 and epochs 150. The batch size is the number of training samples that the model processes before updating the model parameters. The epocs are the number of times that the model iterates over the entire training data.

Subsequently, the predictions are generated, and the confusion matrix, loss, and accuracy are calculated. For the low-level quantities the accuracy is **0.5553**, the loss is **0.9633** and the confusion matrix is the following.

$$\begin{bmatrix} 401 & 387 \\ 325 & 488 \end{bmatrix}$$

For the high-level data, the accuracy is **0.6989** and the loss is **0.5949**. The confusion matrix is:

$$\begin{bmatrix} 475 & 313 \\ 169 & 644 \end{bmatrix}$$

# 6 Discussion

As we can see from the results above, the accuracy of all the methods employed is relatively low, around 0.6. For the low-level data, all results are lower compared to the high-level data. This is partly because the signal in the low-level data is not as distinguishable from the background as it is in the high-level data.

The accuracy for the low-level quantities is similar across the methods, with the ANN approach being slightly lower. This might result from the lower number of neurons added to the network compared to the inputs.

The high energy quantities achieve a better accuracy score with the ANN approach, making it the most effective method. The high-level quantities achieve a better accuracy score than the low-level, with the ANN approach performing the best.

# References


[1] P. Baldi, P. Sadowski, and D. Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature Communications*, 5(1), July 2014.

[2] Diakonidis Thodoros. Lecture notes on machine learning, June 2024. Unpublished.