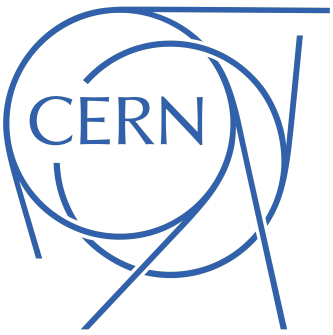


Summer Student Project 2021 Report

Primary Vertex Selection in Higgs to 2 Photon Events

Chih-Chun Kuo
National Taiwan University, Taipei, Taiwan

Supervisors:
Anthony Morley, Valentina Cairo

August 27, 2021

Abstract

Determining the position of the primary vertex that produces a Higgs boson is critical to maximize our sensitivity to a Higgs signal. In $H \to \gamma\gamma$ decays the vertex reconstruction allows us to calculate the photons direction accurately, and hence the di-photon invariant mass resolution. The challenge is to select the correct one among the multiple possible positions. In our work, we try to use some machine learning methods to improve the accuracy and efficiency of selection. Furthermore, this project will investigate new options that will be optimal now (Run 2/Run 3) and in the future (HL-LHC) using additional information from the reconstructed event. So far, when using the random forest plus AdaBoost, we get the best accuracy rate of 96.5%.

Contents

1 Introduction

After four-year operation for the ATLAS[2], beams in the Large Hadron Collider came to a stop in December, 2018. Run 2 was an extraordinary resource for physicists, allowing them to perform the most optimal exploration of the 13 TeV energy frontier.

Over the course of Run 2, ATLAS recorded about 150 fb^{-1} of proton–proton collision data of which 94% are good for physics analysis, compared to 25 fb^{-1} recorded in Run 1. Thus, study of the famed Higgs boson was especially fruitful during Run 2.[1]

1.1 Aim and motivation of the works

$H \to \gamma\gamma$ plays an important role in the ATLAS experimental program because its signal to background ratio is good and it has excellent two photon invariant mass resolution.[3] In the decay analysis, the reconstruction of the primary vertex is critical to maximize our sensitivity to a Higgs signal. So our objective is to select the correct one among the multiple possible positions.

There are several reasons that make this task challenging: First, photons are neutral particles, so we can't directly make the reconstruction of the position of the hard interaction of their tracks.[3] Even, most of the photons will convert into e^+e^- pairs. Thus, we have to reconstruct the position of photons either from their clustering positions or from conversion tracks. Moreover, it is also necessary to discriminate between the hard primary vertex and the vertices originated from concurrent soft collisions (pile-up vertices).[3] As you can see, there exists a lot of experimental parameters in the process , which are what we are going to focus on.

As a result, the work will be done using simulated events of Run 2 environment and we've tried several machine learning methods to improve the efficiency and accuracy of identifying primary vertex in $H \to \gamma\gamma$ events.

1.2 Analysis steps

The work is divided in the following steps:

- Pre-processing
  - Understanding relations between features
  - Selecting useful features
  - Labeling data
- Machine learning
  - Random forest
  - Random forest + AdaBoost
- Post-processing

2 Software

During the analysis the following software have been used.

2.1 ROOT

ROOT[4] is an object-oriented framework aimed at solving the big data analysis challenges of high-energy physics. It is written mainly in C++ and can be used on Linux, macOS, or Windows. ROOT also integrates super-smoothly with Python thanks to its unique dynamic and powerful Python $\rightleftharpoons$ C++ binding. In this project, we us ROOT to do basic data analysis.

2.2 Scikit-learn

Scikit-learn[5] is an open source Python machine learning library that supports supervised and unsupervised learning. It also provides various tools for model fitting, data pre-processing, model selection and evaluation, and many other utilities. In this project, we us Scikit-learn to do classification.

3 Pre-processing

3.1 Experimental parameters

A simple event diagram with the experimental parameters is shown in Figure. 1.

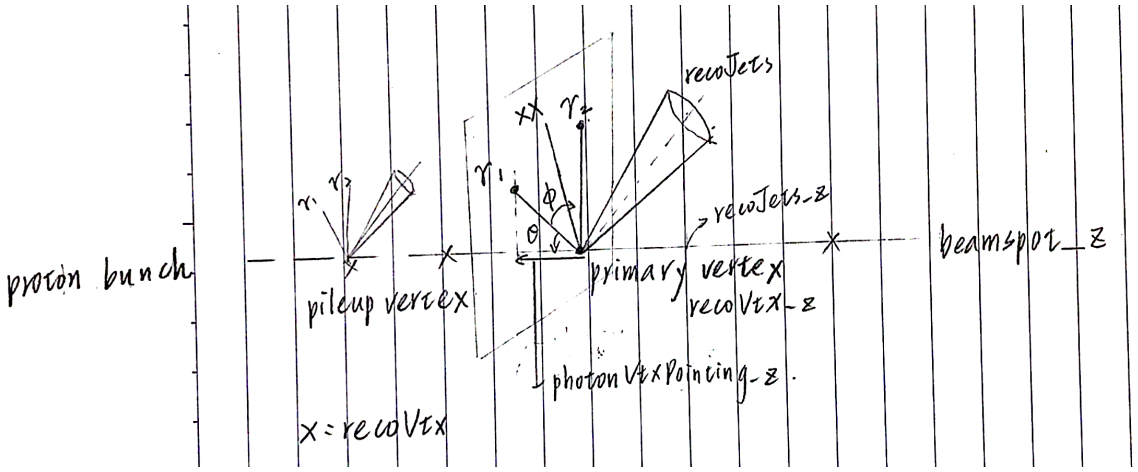


Figure 1: Event diagram (note that γ_1, γ_2 represent the same two photons)

- General parameters

  -`pileup`: The number of pp collisions in the event.

  -`true_z`: The true position of the collision. (from MC)

  -`true_z1`: The true origin the first photon. (from MC)

  -`true_z2`: The true origin the second photon. (from MC)

  -`beamspot_Z`: The center of the interaction region.

  -`beamspot_Z_width`: The length of the interaction region.

  -`XX_m`: The mass of the 2 photons (MeV).

  -`XX_pt`: The transverse momentum of the 2 photon system (MeV).

  -`XX_eta`: The direction of 2 photon system.

  -`XX_phi`: The direction of 2 photon system.

- Hadronic jets

  -`recoJets_pt`

  -`recoJets_eta`

  -`recoJets_phi`

  -`recoJets_m`: Mass of the jet.

  -`recoJets_z`: Average z position of the tracks associated to the jet.

  -`recoJets_z_rms`: RMS of the z position of the tracks associated to the jet

  -`recoJets_z_w`: Weight average z position of the tracks associated to the jet.

  -`recoJets_z_rms_w`: RMS of the weighted z position of the tracks associated to the jet.

  -`recoJets_nTrks`: Number of tracks matched to the jet.

- Reconstrucetd primary verticies

  -recoVtx_nTracks: Number of tracks associated to the vertex.

  -recoVtx_z: Z position of the vertex.

  -recoVtx_z_err: Unertainty in the Z position pf the vertex.

  -recoVtx_pt

  -recoVtx_phi

  -recoVtx_eta

  -recoVtx_theta

  -recoVtx_sumpt

  -recoVtx_sumpt2

  -recoVtx_truthMatchType=(vector<int>*): Truth properties of the vertex.

  -enum VertexMatchType{ MATCHED: > threshold (default 70%) from one truth interaction, MERGED: not matched, SPLIT: highest weight truth interaction contributes to > 1 vtx (vtx with highest fraction of sumpT2 remains matched/merged), FAKE: highest contribution is fake (if pile-up MC info not present those tracks end up as "fakes"), DUMMY: dummy vertex, NTYPES:number of types }

  -recoVtx_truthHStype=(vector<int>*): Truth properties of the vertex.

  -enum HardScatterType{ CLEAN: matched, LOWPU: merged, but dominant contributor, HIGHPU: merged and dominated by pile-up or fakes, HSSPLIT: split, NONE: not found, NHSTYPES }

  -recoVtx_dphi: Difference in direction w.r.t di-photon system.

  -recoVtx_dpt: Difference in transverse momentum w.r.t di-photon system.

  -recoVtx_dphi_jet: Difference in direction compared to the jets associated to the vertex.

  -recoVtx_dpt_jet: Difference in momentum compared to the jets associated to the vertex.

  -recoVtx_njet: Number of jets associated to the vertex.

  -recoVtx_dphi_jetFwd: Difference in direction compared to the jets associated to the vertex including fwd jets.

  -recoVtx_dpt_jetFwd: Difference in momentum compared to the jets associated to the vertex including fwd jets.

  -recoVtx_njetFwd: Number of jets associated to the vertex including fwd jets.

- Photons

  -cluster_Pt

  -cluster_Eta

  -clusterPointing_z

  -clusterPointing_z_err

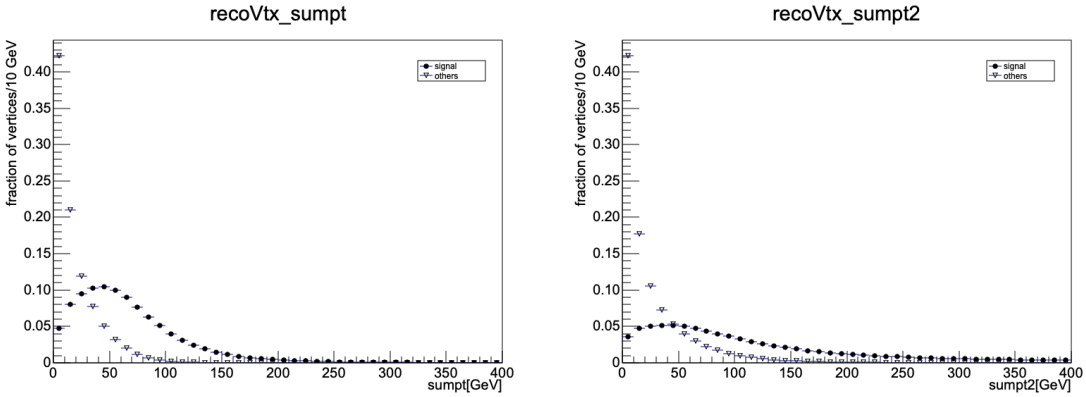
  -photonVtxPointing_z: Z position from cluster-conversion vertex information.

  -photonVtxPointing_z_err: Z position uncertainty from cluster-conversion vertex information.

  -photonTrkPointing_nTrk: Number of tracks in the conversion vertex.

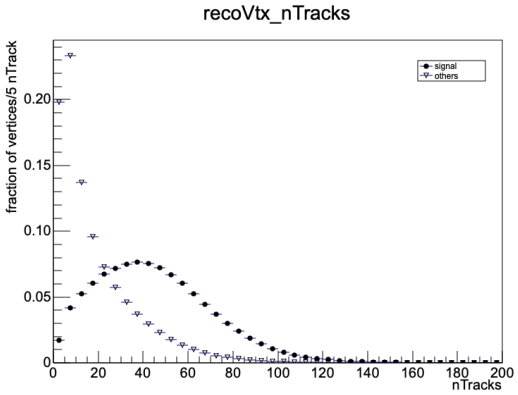
  -photonTrkPointing_z: Z position from conversion vertex information.

  -photonTrkPointing_z_err: Z position uncertainty from conversion vertex information.

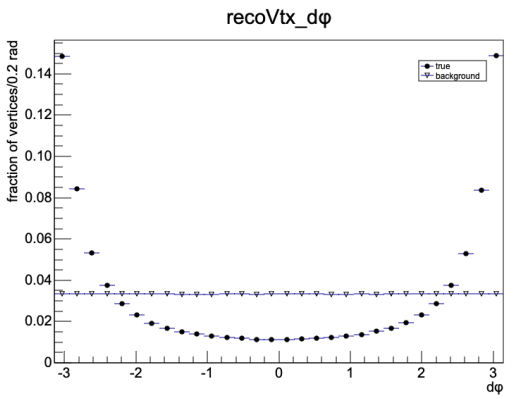


(a) Σp_T

(b) Σp_T^2



(c) nTracks



(d) $d\phi$

Figure 2: Feature distribution of true vertices and background

3.2 Feature selection

In this section, we identify useful features from parameters in Section. 3.1 by sense of physics and checked if such feature distributions of true vertices are different from that of background vertices, to avoid time- and memory-wasting in the learning process. To accomplish this, we separately plot some of the physical quantities we consider to be more distinguishable between signal and background, and observe their distribution. These features are shown in Figure. 2.

We can see that at certain values, the occurrence of true vertices is higher. The lower energy of the background is mostly due to soft QCD interactions of the protons, which are less than those able to produce a higgs boson, and that is why they have less tracks. The bigger $|d\phi|$ of the vertex suggests the di-photon system is recoiling from the other remnant of the proton collision or vice-versa. Such a discrimination will be useful in the decision tree. Hence, we put them in to the classifiers and perform training. The actually useful features for random forest classifier can be identified from the function of the feature importance, which is shown in Figure. 3. These features are proved to be useful. But one other thing worth noting is that the parameters we have used so far is just an attempt, for completeness or for improvement, we still have to add other might-be useful parameters to the classifier.

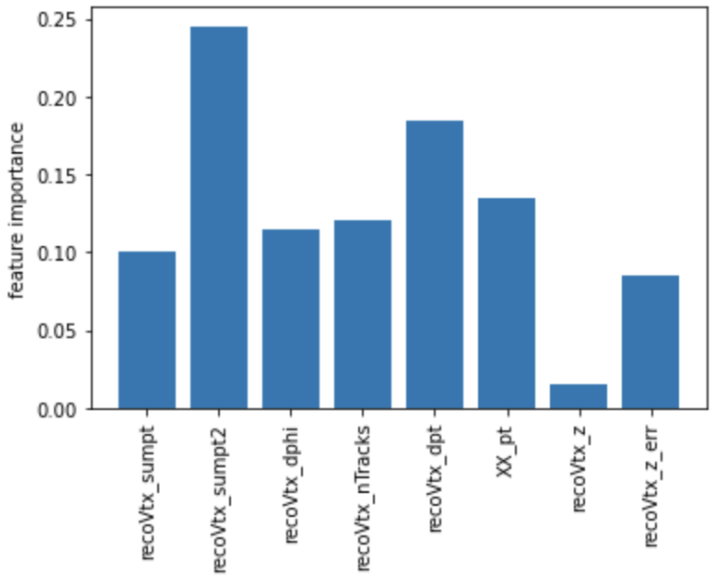


Figure 3: Feature importance

3.3 Data labeling

We will use classification in our work. So before training, our first step is to label our data, which means classifying all vertices in the event as either signal or background. Then after doing this, the classifier will be trained to predict the testing data or in the future select the most likely vertex in the event to be the signal vertex.

In data labeling, first we split each vertex in the same event into independent samples and keep the original parameter `entry` to record the event to which the sample belongs.[Figure. 4] Second we use 0 and 1 to label each sample, where 0 represents background vertex and 1 represents true vertex. More specifically, we label the vertex closest to the Monte Carlo simulation result (`true_z`) in each event as true vertex.

	entry	recoVtx_sumpt	recoVtx_sumpt2	recoVtx_dphi	clusterPointing_z1	clusterPointing_z2	photonVtxPointing_z1	photonVtxPointing_z2	recoVtx_nTracks
0	0	1.633276	1.767972	2.091142	-21.095335	4.450816	0.0	0.0	41
1	0	1.390695	1.669842	2.412117	-21.095335	4.450816	0.0	0.0	20
2	0	1.531884	1.651738	-1.347121	-21.095335	4.450816	0.0	0.0	32
3	0	1.490315	1.441895	2.912775	-21.095335	4.450816	0.0	0.0	39
4	0	1.167896	1.236481	1.785630	-21.095335	4.450816	0.0	0.0	16
5	0	1.077219	1.042398	-0.730479	-21.095335	4.450816	0.0	0.0	15
6	0	1.005396	0.958269	2.280881	-21.095335	4.450816	0.0	0.0	13
7	0	0.854982	0.953054	1.126697	-21.095335	4.450816	0.0	0.0	8
8	0	0.824721	0.935388	-2.353534	-21.095335	4.450816	0.0	0.0	6
9	0	0.772029	0.692645	2.908410	-21.095335	4.450816	0.0	0.0	8
10	0	0.804503	0.674586	-0.839742	-21.095335	4.450816	0.0	0.0	9
11	0	0.612453	0.448598	1.026081	-21.095335	4.450816	0.0	0.0	6

Figure 4: Data after pre-processing

4 Random forest

4.1 Concept

This supervised learning procedure operates according to the "divide and conquer" principle: sample fractions of the data, grow a randomized tree predictor on each small piece, then paste (aggregate) these predictors together into a forest. With using Gini factor, each tree decides the sample either true or false, and then the forest votes for the prediction.[6]

There are two kinds of trees to build up a random forest : imbalanced trees and balanced trees. Imbalanced tree means, all classes are supposed to have weight one. In our case, class 0 accounts for about 95% while class 1 accounts for 5%. On the other hand, balanced tree uses the values, which are inversely proportional to class frequencies in the input data, to adjust weights.

4.2 Implementation and result

For analysis, we use three indicators to distinguish quality of models: confusion matrix, balanced accuracy and normal accuracy (imbalanced accuracy). The formula for calculating balanced accuracy is:

$$\frac{\text{accuracy(class 0)} + \text{accuracy(class 1)}}{2}$$

First we tune the hyper-parameter of the trees in random forest. We have tried some parameters under the constraints of memory and time. The following parameters are most suitable for the current situation but not optimal. We set `min_samples_leaf=0.1`, which means the minimum number of samples required to be at a leaf node is 0.1 times the size of the training set. We implement these imbalanced trees and adjust the value of t `n_estimators`, but the imbalanced accuracy is limited to 95%.[Figure 5] From the confusion matrix we know that the classifier always labels our testing set as class 0, so implementing balanced trees is imperative. Repeating the above actions and we see the accuracy will reach saturation at `n_estimators=400` but less than 80%.[Figure 5]

5 Random forest + AdaBoost

5.1 Concept

Boosting is an approach to machine learning based on the idea of creating a highly accurate prediction rule by combining many relatively weak and inaccurate rules. Pseudocode for AdaBoost is shown below[7] :

Listing 1: AdaBoost Pseudocode

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in \mathcal{X}$, $y_i \in \{-1, +1\}$.
Initialize: $D_1(i) = 1/m$ for $i = 1, ..., m$.
For $t = 1, ..., T$:

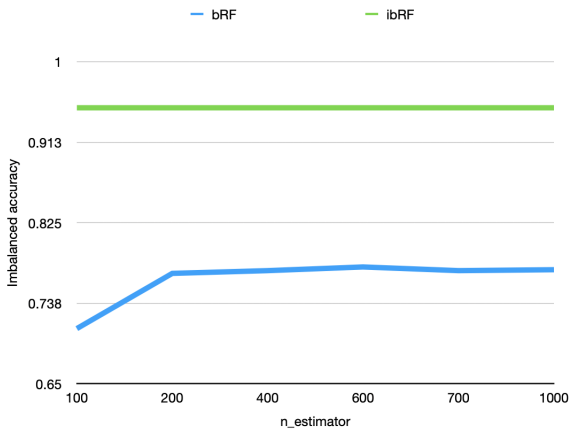


Figure 5: Imbalanced accuracy of imbalanced and balanced random forest

Train weak learner using distribution D_t.
Get weak hypothesis $h_t : \mathcal{X} \to \{-1, +1\}$.
Aim: select h_t with low weighted error: $\epsilon_t = P_{i \sim D_t}[h_t(x_i) \neq y_i]$.
Choose $\alpha_t = \frac{1}{2} ln(\frac{1-\epsilon_t}{\epsilon_t})$.
Update, for $i = 1, ..., m : D_{t+1}(i) = \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t}$.
Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).
Return the final hypothesis:
$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$

5.2 Implementation and result

We use random forest again as the weak learner and set $T = 100$. Since it is a weak learner, we set `n_estimators` of the random forest as 2, which means building two trees in one forest, and set `max_depth` as 1, which means a tree has a single node. Again, this is the most suitable hyper-parameters so far, but not optimal. The prediction percentage of each actual class and the number of samples are shown in Table. 1.

	PREDICTED NEGATIVE	PREDICTED POSITIVE
ACTUAL NEGATIVE	6774970 (85.3%)	1167228 (14.7%)
ACTUAL POSITIVE	75169 (18%)	338676 (82%)

Table 1.A Balanced tree

	PREDICTED NEGATIVE	PREDICTED POSITIVE
ACTUAL NEGATIVE	7906940 (99.6%)	35258 (0.4%)
ACTUAL POSITIVE	263437 (63.3%)	150408 (36.7%)

Table 1.B Imbalanced tree

Table 1: Confusion matrix

So for balanced tree: the imbalanced accuracy is **85.1%** and the balanced accuracy is **83.6%**, for imbalanced tree: the imbalanced accuracy is **96.4%** and the balanced accuracy is **67.9%**, which are an improvement of just using random forest.

6 Post-processing

After training several classifiers of random forest+AdaBoost, there are some ways to do ensemble.

6.1 Adding up the probability of each classifier

In this way, we let the output of the prediction be probability of each class and add them up sample by sample. Then, the class with higher probability will be the final prediction. Another thing worth mentioning is that there isn't any sample with same probability of each class.

We use two balanced tree classifiers and two imbalanced tree classifiers to do ensemble and get the imbalanced accuracy of **94.9%** and balanced accuracy of **77.5%**.

6.2 Voting by three classifiers

When using two imbalanced tree classifiers and one balanced tree classifier, the imbalanced accuracy comes to **96.4%** and balanced accuracy comes to **69.3%**.

When using one imbalanced tree classifier and two balanced tree classifiers, the imbalanced accuracy comes to **86.5%** and balanced accuracy comes to **83.5%**.

6.3 Voting by two classifiers

In this way, we use one balanced tree classifier and one imbalanced tree classifier for ensemble. Let the first digit be the result of balanced tree and the second digit be that of the imbalanced tree. The possible set of the result becomes {00,01,10,11}. After some simple statistics, {01} doesn't exist, so the set of the result becomes {00,10,11}. And the fact that 01 doesn't exist is reasonable.

Since the entry feature, used to label which vertex belonging to which event, is useless in random forest, which can be seen by the feature importance function of the classifier in scikit-learn, manually post-processing is needed.

Step 1. If the result is {00}, then the final result is class 0.

Step 2. If the result is {11}, it is highly possible that the vertex is true vertex. Since there is only one true vertex in each event, we have to understand how the result distributes by event first. In this testing set, there is 56.7% with zero {11}, 41.1% with one {11}, 2.1% with two {11}, 0.1% with three {11}. Sine the number of {11} in each event is less than four (the number of reconstruction vertices in one event ranging from 0 to 50) and events with more than one {11} account for a very small part , it is acceptable to do following things : If the event has more than one {11}, then the final result of all {11} vertices becomes 1 and for {10} vertices in this kind of event, the final result is suppressing to 0.

Step 3. Now the problem is, there is 56.7% of the events that doesn't have {11} vertices.

The number of {10} in this case can be seen from Figure. 6:

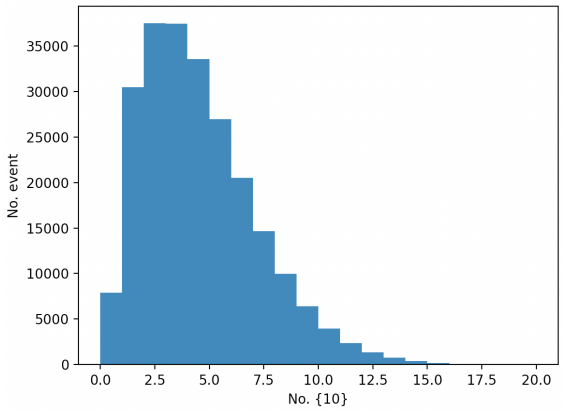


Figure 6: Imbalanced accuracy of imbalanced and balanced random forest

We have some different choices now :
case 1. Let 10 in these events be 0
case 2. Let 10 in these events be 1
In case 1, imbalanced accuracy comes to **96.5%** and balanced accuracy comes to **68.4%**. In case 2, imbalanced accuracy comes to **89.1%** and balanced accuracy comes to **83.8%**.

7 Comparison and Conclusion

The results of the various methods described above are shown in the Table. 2. So far, the two indicators have not been able to achieve the best at the same time. But overall, the results after manual post-processing are better. In the future, detailed figures of decision trees from the classifiers can be used to in-depth understand the use of different features in decision-making, in order to improve the accuracy. The example decision tree diagram is shown in Figure. 7.

Method	Balanced Accuracy	Imbalanced Accuracy
Add up probability	77.5%	94.9%
Three classifiers vote	69.3%	96.4%
Two classifiers vote (case 1)	68.4%	96.5%
Two classifiers vote (case 2)	83.8%	89.1%

Table 2: Results

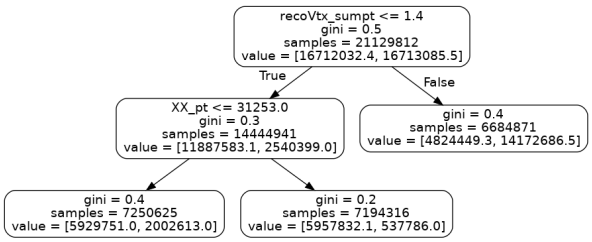


Figure 7: Tree of random forest

8 Acknowledgement

First of all, thank you very much CERN for providing such an opportunity, allowing me to learn about various accelerators, detectors, particle physics, and experience the life of full-time research in just two months.

Secondly, and most importantly, thank you Anthony and Valentina for their care and guidance, helping me get into this environment and complete the research topics. Thank you for being always happy to answer my questions. The weekly meeting not only allows me to find new directions for improvement, it is also a very pleasant experience to communicate with people who span half of the world about each other's life.

Thank you for allowing me to spend a very fulfilling and happy summer vacation. I also hope that I will be more outstanding in the future and have the opportunity to work with you together again.

References

[1] Katarina Anthony: ATLAS completes data-taking for Run 2, `https://atlas.cern/updates/news/atlas-completes-data-taking-run-2`

[2] ATLAS Collaboration. *The ATLAS Experiment at the CERN Large Hadron Collider*. JINST 3 (2008) S08003

[3] Andrea Visibile. *Determination of the primary vertex in di-photon events with the ATLAS detector Relatore: Correlatori: at the LHC*. "Bachelor's Thesis".

[4] Rene Brun and Fons Rademakers. *ROOT - An Object Oriented Data Analysis Framework*. Proceedings AIHENP'96 Workshop, Lausanne, Sep. 1996, Nucl. Inst. & Meth. in Phys. Res. A 389 (1997) 81-86. See also "ROOT" [software], Release v6.20/04, 23/08/2019.

[5] Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, E. *Scikit-learn: Machine Learning in Python*. , Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[6] Gérard Biau and Erwan Scornet. *A random forest guided tour*. TEST, 2016.

[7] Bernhard Schölkopf, Zhiyuan Luo and Vladimir Vovk. *Empirical Inference*. Springer-Verlag, Berlin Heidelberg, 2013.