Contents

1. Introduction

1.1. Motivation

Goal of this project is the improvement in the distinction process between $t\bar{t}\gamma$ and $t\bar{t}$ events in the signal region. Depending on the success, the results can be further used for other projects, which would improve the simulations and their predictions regarding the Standard Model (SM) and Beyond Standard Model (BSM).

1.2. Approach

An attempt was perfomed through a multivariate analysis by developing and training models. At the beginning different variables were analysed and compared and the most promising were used for training. The type of models, denoted MVA in the following, used for this project are boosted decision trees (BDT) and mutilayer perceptron (MLP). The results of the MVAs for three different variable categories were created and in the course of this project analysed and presented.

2. Background information

2.1. ATLAS detector and data

The ATLAS detector is part of the Large Hadron Collider, a particle accelerator, which was built by the research organisation CERN in Geneva, Switzerland. LHC has a circumference of 27km and is up to 175m deep. Besides ATLAS, there are other experiments as well, such as CMS, ALICE and LHCb. ATLAS has three sub-detectors, which all surround the beam pipe, where the particles are accelerated and collided against eachother.

In Figure 1, a sketch of the detector system can be seen. The inner detecor is a tracking chamber for charged partices. It is surrounded by a superconducting solenoid with a magnetic field of 2T used to divert the particle trajectory to measure the momentum. The calorimeter is made up of alternating layers. One is a dense absorber material, which is used to prompt electromagnetic and hadronic showers (ECAL, HCAL). The other layer is of active material used for energy measurement. The muon spectrometer (MS) is positioned in the outermost layer since muons may pass inner layers as well as the calorimeters. It is surrounded by a magnetic field and the diversion is measured by a multiple layer of high precision tracking chambers [1, 2].

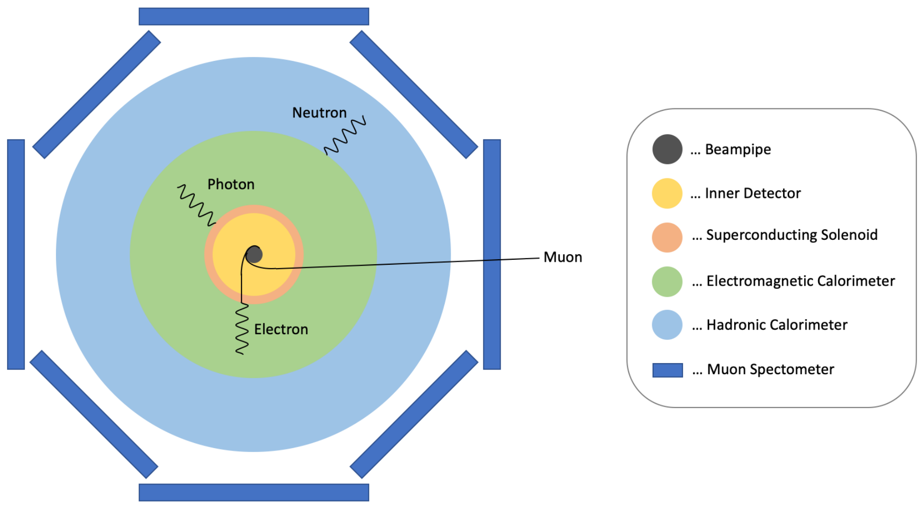


Figure 1: Detector system of ATLAS

The data is filtered by a two-step triggersystem. The hardware-based system (L1) is triggered when muons, electrons, photons and jets are detected. This reduced the data amount from 40MHz to 100kHz. The software-based system (L2) is triggered by the different energies, type and numbers of particles:

- lepton has $30/35$ GeV
- at least 3 jets were detected
- at least one of the jets have a b-tag
- exactly one photon

This reduced the data amount from 100kHz auf 1kHz. The data used during this project was recorded in 2016 at a centre-of mass energy of 13TeV. The signal and background processes were modelled using Monte Carlo generators and passed through a detector simulation [1].

2.2. Proton-Proton-collision and $t\bar{t}\gamma$

The protons are accelerated in bunches, each containing 10^{11} particles, through different beam pipes in opposite directions and then collided with each other. During the pp-process a top quark pair is produced. The top quark then decays to a W-boson and a b-quark. The W-boson disintegrates into a charged lepton and a corresponding neutrino or a pair of up-type quark and down-type antiquark. The remaining quarks decomposes into jets. This process can be seen in Figure 2.

The $t\bar{t}\gamma$ events were selected - top quark pair events with additional photons. The top quark is the heaviest quark in the known elementary particles and is the only quark so far whose properties can be analysed since their lifetime is very short and it forms no bound state before it decays. By that the electromagnetic coupling, therefore the number of radiated photons after interaction with other charged particles, can be measured [2]. By analysing the top quark, the results can be used to test the Standard Model and its possible extensions [1].

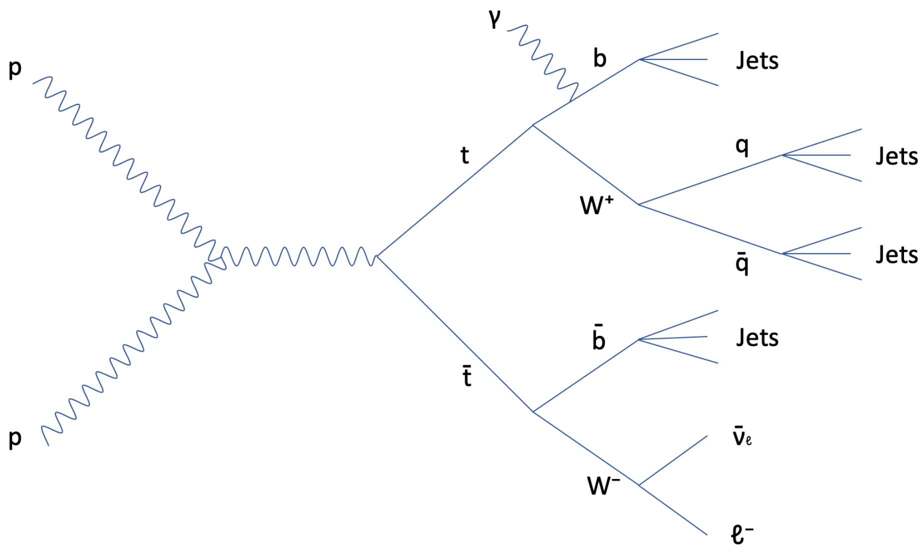


Figure 2: Proton Proton collision process

3. Challenge in categorisation

In Figure 3, the transverse momentum of the photon is visible. The background makes up nearly 50% of the signal region. The possible sources are [1, 2]:

- hadron or electron misidentified as photon
- misconstructed photon
- photon radiated from different process
- additional particles not detected due to limited detector's acceptence
- hadron misconstructed as electron

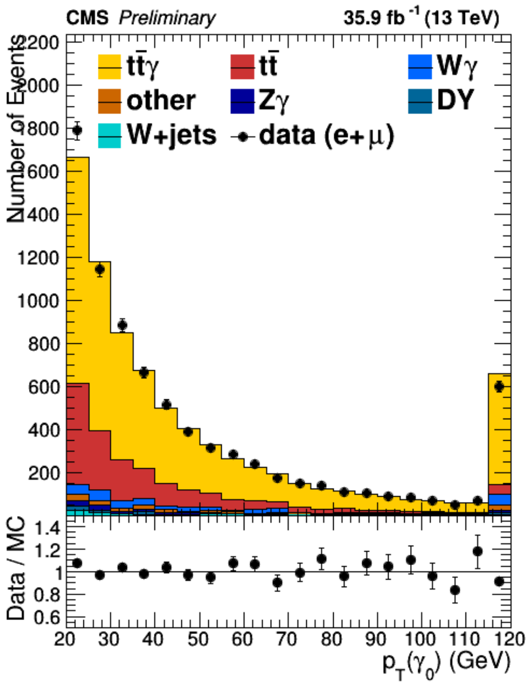


Figure 3: Distribution of P_T Photon

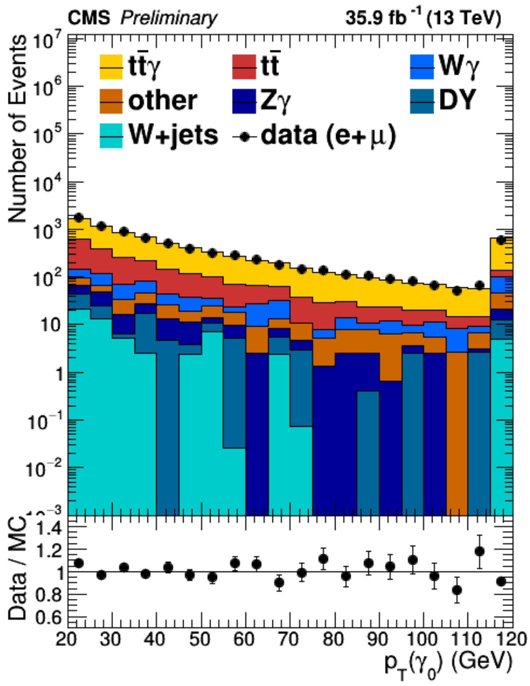


Figure 4: Distribution of P_T Photon (log)

3.1. Current status

First a neural network was developed to seperate the hadronic-fake photons, which are hadrons misidentified as photons and made up the majority of the background signal. The result was then used as an input for the neural network created to split the $t\bar{t}\gamma$ signal and the background further [1]. What still needs to be investigated is the possibility of improvement in the discrimination between $t\bar{t}\gamma$ and $t\bar{t}$ events in the signal region through MVAs.

4. Analysis process

4.1. List of variables

The analysed variables are listed below and they are categorized by their type of particles.

- B-jet: variables of jets containing b-hadrons (b-tag)
  - Bj0_eta
  - Bj0_phi
  - Bj0_pt
  - Bj1_eta
  - Bj1_phi
  - Bj1_pt
  - nBTagGood

- Jet: variables of all jets
  - JetGood0_eta
  - JetGood0_phi
  - JetGood0_pt
  - JetGood1_eta
  - JetGood1_phi
  - JetGood1_pt
  - nJet
  - nJetGood
  - JetGood0_neEmEF
  - JetGood0_chEmEF
  - JetGood0_neHEF
  - JetGood0_chHEF
- Lepton: variables of electron, muon
  - LeptonTight0_eta
  - LeptonTight0_phi
  - LeptonTight0_pt
  - nElectronTight
  - nLeptonTight
  - nMuonTight
  - LeptonTight0_pfRelIso03_all
  - LeptonTight0_pfRelIso03_chg
  - LeptonTight0_pfRelIso03_ne
- Combination: variables of two particle types
  - leptonJetdR
  - ltight0GammadPhi
  - PhotonLepdR
  - PhotonJetdR
  - JetGood0_btagDeepB
  - JetGood1_btagDeepB
- Photon:
  - nPhotonGood
  - PhotonGood0_eta
  - PhotonGood0_mvaID
  - PhotonGood0_phi
  - PhotonGood0_pt
- Others:
  - MET_phi
  - MET_pt
  - ht
  - m3
  - mT

The pre- or suffix of the name reveal what information is saved. The description of variables, which don't follow this logic, are listed at the end.

Prefix and suffix:

- _eta: pseudorapidity ($\eta = -ln(tan(\frac{\vartheta}{2}))$; wheras ϑ is the polar angle)

- _phi: azithmuthal opening angle around z-axis ($\Delta\phi$)
- _pt: transverse momentum of variable
- dR: distance between two objects (ΔR)
- n: number of particles

Other:

- ht
- m3
- mT
- PhotonGood0: number of photons
- PhotonGood0_mvaID: currently used variable to differentiate between signal and background events
- MET: Missing transverse momentum
- JetGood0_neEmEF
- JetGood0_chEmEF
- JetGood0_neHEF
- JetGood0_chHEF
- LeptonTight0_pfRelIso03_all
- LeptonTight0_pfRelIso03_chg
- LeptonTight0_pfRelIso03_ne
- JetGood0_btagDeepB
- JetGood1_btagDeepB

4.2. Analysis of variables

The variables were analysed by comparing the distribution of events with an uniform scaling. An uniform scaling was needed to have a better view of relative differences between signal and background. If the variable shows a different behaviour for each type of signal, this could be an indication of discriminatory power.

Figure 5 to 8 show some examples of variables, where the distributions are almost identical and were therefore not further analysed. A full list of these variables are listed in the Annex, chapter A.1.

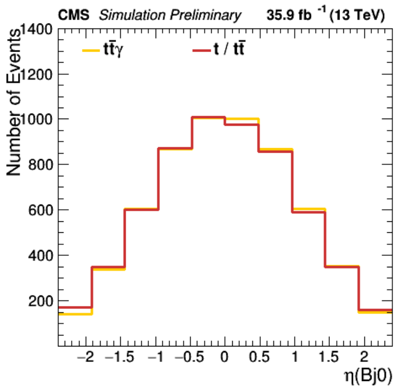


Figure 5: Bj0_eta

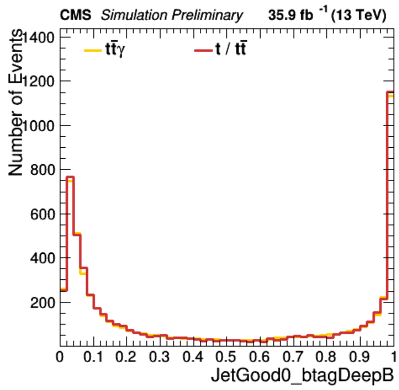


Figure 6: JetGood0_btagDeepB

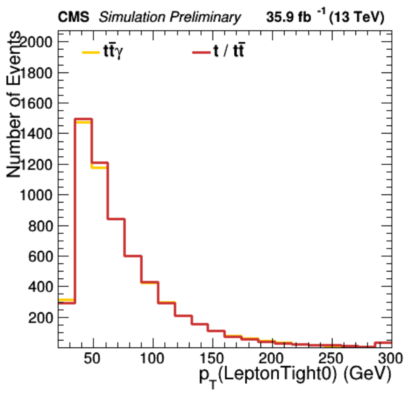


Figure 7: LeptonTight0_pt

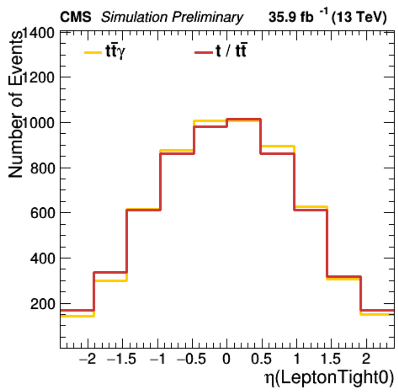


Figure 8: LeptonTight0_eta

Variables, where there are some differences visible in the signal region are in Figure 9 to 19. These were candidates for the first set of selected variables.

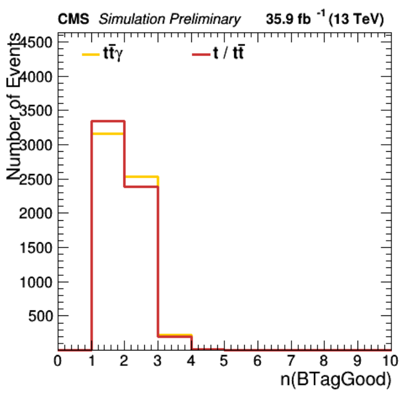


Figure 9: nBTagGood

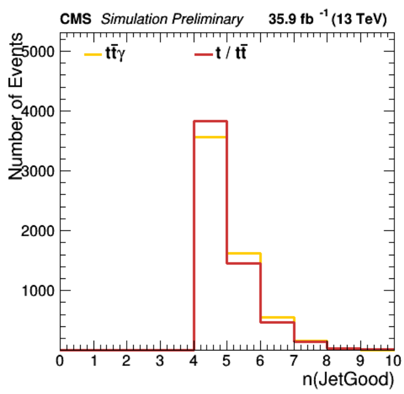


Figure 10: nJetGood

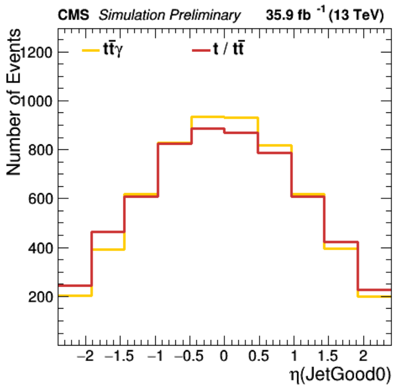


Figure 11: JetGood0_eta

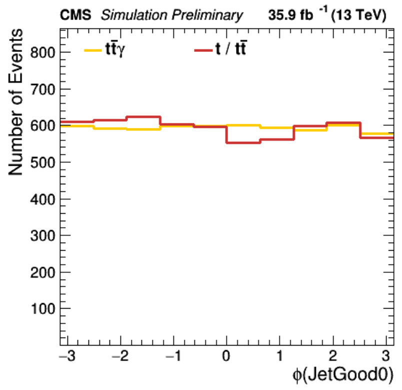


Figure 12: JetGood0_phi

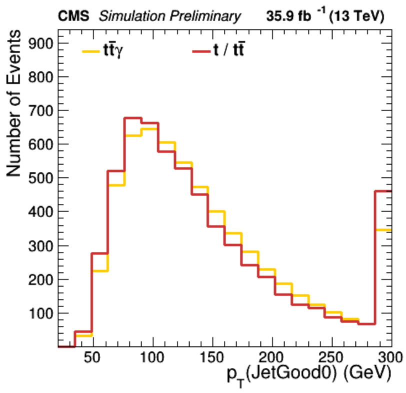


Figure 13: JetGood0_pt

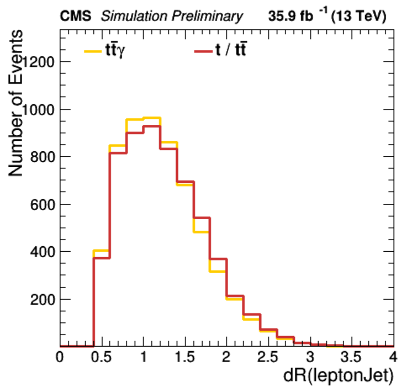


Figure 14: leptonJetdR

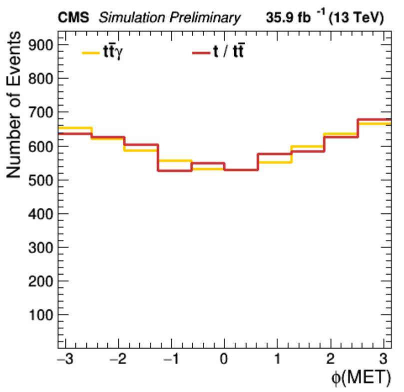


Figure 15: MET_phi

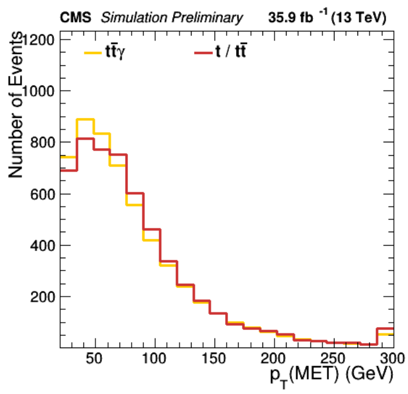


Figure 16: MET_pt

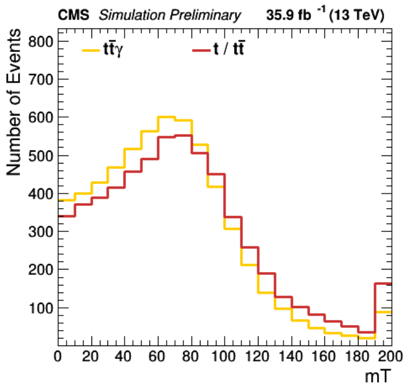


Figure 17: mT

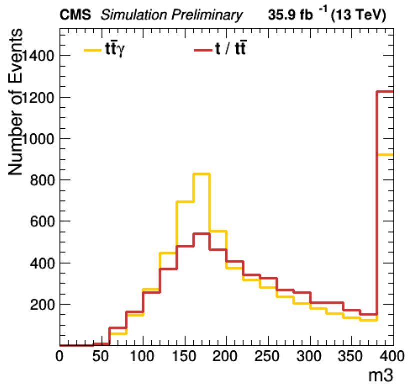


Figure 18: m3

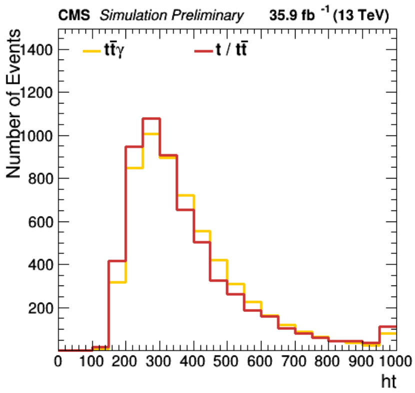


Figure 19: ht

The variable PhotonGood0_mvaID, seen in Figure 20, shows quite a good distinction and is therefore added to the list (second selection).

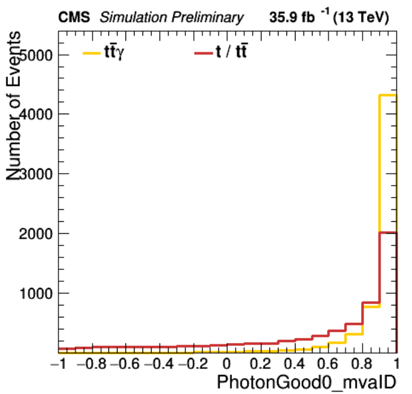


Figure 20: PhotonGood0_mvaID

Figure 21 to 26 show the photon variables. These show the most promising behaviour for discriminating between $t\bar{t}\gamma$ and $t\bar{t}$ events and were put additionally into the third selection category.

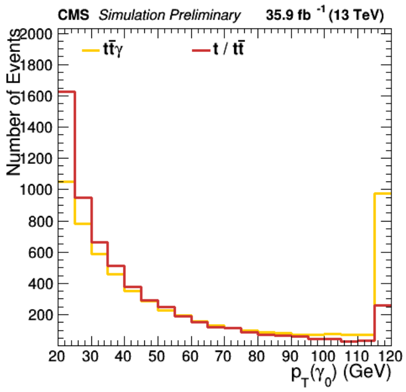


Figure 21: PhotonGood0_pt

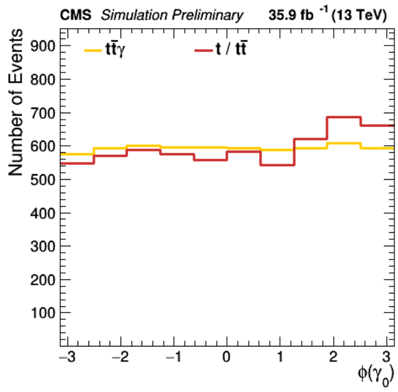


Figure 22: PhotonGood0_phi

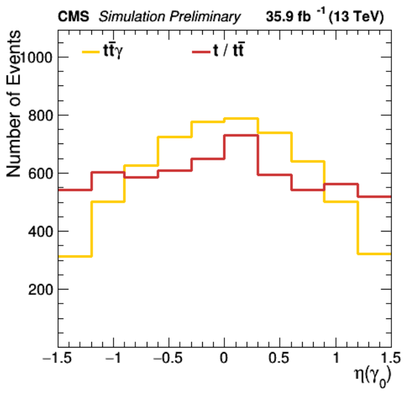


Figure 23: PhotonGood0_eta

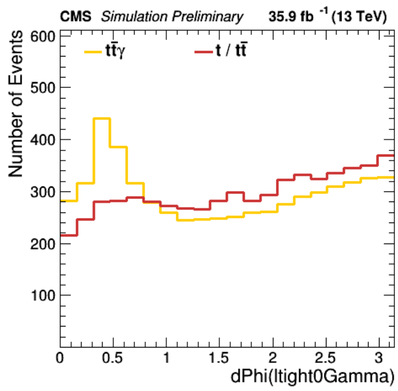


Figure 24: ltight0GammadPhi

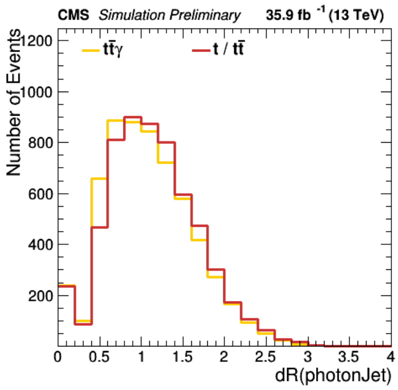


Figure 25: photonJetdR

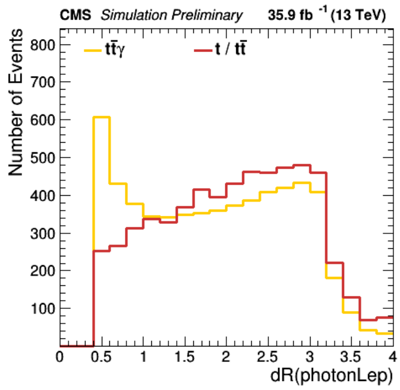


Figure 26: photonLepdR

5. MVA

Trying to split signal and background through a variable with a cut-off value faces the problem that this criteria doesn't apply for every event. If it works for one event, it doesn't mean that it works for the next one. Therefore mutliple variables need to be used to distinguish between those two categories, so called multivariate analysis (MVA). As already mentioned in chapter 1.2, the chosen MVA-types are boosted decision trees (BDT) and multilayer perceptron (MLP).

5.1. BDT - Boosted Decision Trees

The decision tree starts with the root node containing the whole sample. For a binary tree, the node is then split into two branches using a variable and a corresponding cut-off value. The cut-off value should be a value, which seperates the signal from the background the best. This process will be repeated for each branch for all relevant variables, including already used variables, until an end criteria is met. The final branch is called a leaf and this will be assigned to one of the two categories. End criterias could be a minimum size of a leaf, perfect separation, insignificant improvement after split or maximal tree depth. Boosted decision trees are built out of additional trees, so called weak classifieres. They include variables with a low discriminatory power and will be used to improve the main decision tree to a more stable model with a lower error rate [3].

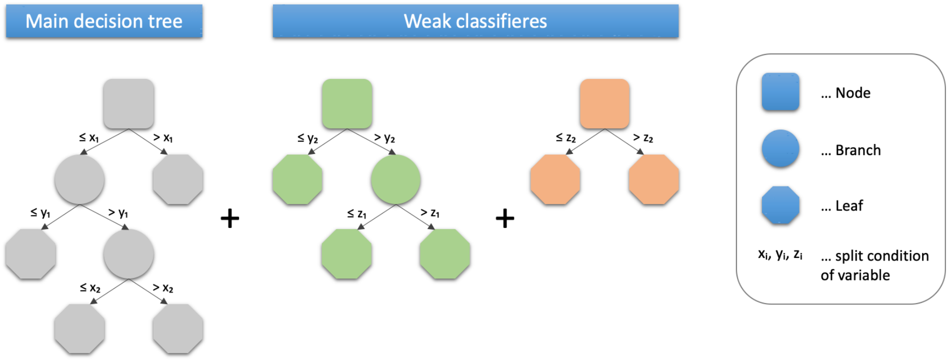
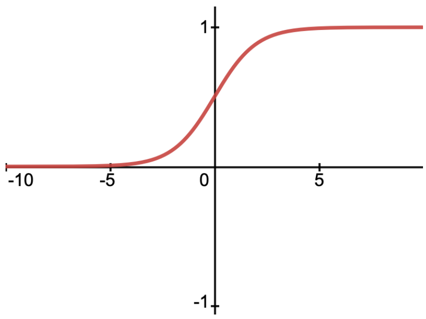


Figure 27: Outline of Boosted Decision Trees

5.2. MLP - Multilayer Perceptron

The model is built out of multiple perceptrons, which are arranged in layers shown in Figure 29. The value of one perceptron is the sum of all weighted inputs. This value is then transformed through an activation function and a treshhold. The activation funtion is a linear function for the input- and outputlayer, while for the hiddenlayers it is usually the log-sigmoid transfer function, seen in Figure 28. All layers have a treshhold, except the inputlayer.



$$f(x) = \frac{1}{1 + e^{-x}}$$

Figure 28: log-sigmoid function

In the MLP, every perceptron of a layer is connected to every perceptron of the previous and next layer. Each connection has an assigned weight, so called weight coefficient. All coefficients of one layer can be written as a weight matrix. The first and last layer are called input- and outputlayer respectively, while the intermediate layers are the hiddenlayers. A MLP has a minimum of one hiddenlayer, therefore contains at least three layers [4].

If the signal is only transmitted in one direction, therefore the MLP doesn't have any loops, it is described as feed forward. The learning process of the model is called backpropagation algorithm. It learns by minimizing the error between network output and expexted output. Initially, all weights are assigned random values, then every event will go through the network and the weights are adapted. Either all events will be put through the network first and then the values are adapted or they are adjusted after each event, but for the latter the order of the events might be important. After one epoch the end criteria will be tested. If it fails, the whole learning process will be carried out again. If it is met, the algorithm is finished [5].

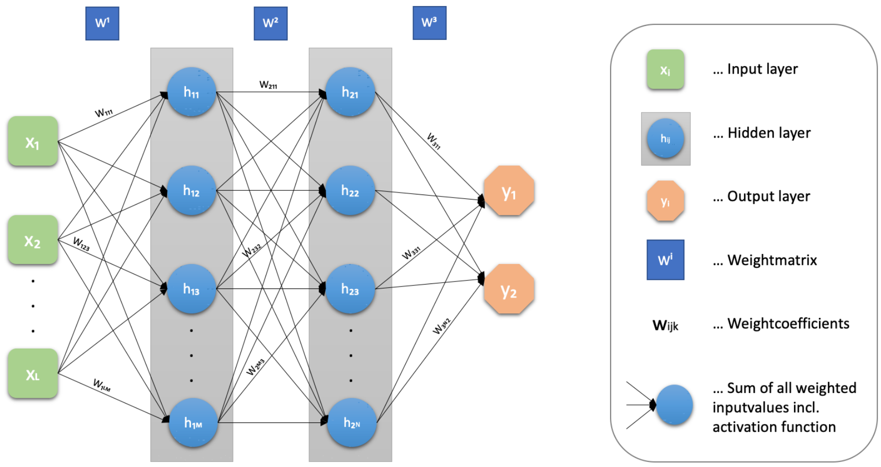


Figure 29: Outline of Multilayer Perceptron

5.3. Training of MVA

The default values of the following configuration options were used for the models:

- BDT
  - Minimum percentage required in a leaf node (MinNodeSize): 5%
  - Separation criterion for node splitting (SeparationType): GiniIndex
- MLP
  - Number of training cycles (NCycles): 500

- ANN learning rate parameter (LearningRate): 0.02
- Decay rate for learning parameter (DecayRate): 0.01
- Neuron input function type (NeuronInputType): sum
- Training Type (TrainingMethod): BP (Back-Propagation)

A full list of all parameters, configuration options as well as their default values are given in the annex, chapter A.2 and A.3 [6].

These parameters were adapted to potentially improve the result:

- BDT
  - Maximum depth of decision trees allowed (MaxDepth)
  - Number of trees in the forest (NTrees)
  - Number of grid points used to find optimal cut (nCuts)
- MLP
  - Number of hidden layers (HiddenLayers)

Their default values were used as a basis, listed in the second row in Table 1. The MVAs were additionally trained with a lower and a higher configuration (row 2 and 3):

	BDT			MLP
Configuration	MaxDepth	NTrees	nCuts	HiddenLayers
1	1	400	10	5
2	3	800	25	7
3	5	1000	50	10

Table 1: Configuration combinations used in MVAs

The distribution and ROC-curves for the BDT- and MLP-models are shown for each selection and configuration in the following chapter. To avoid overfitting, each MVA is trained on one part of the sample (training sample) and tested on the rest (testing sample). If the results differ significantly, then this would be a sign of overfitting - the model only differentiates correctly on this specific sample. The results of the train- and test-sample can be seen in the distribution plots.

The ROC-curves (receiver operating characteristic curve) show the background rejection versus signal efficiency. The AUC (area under the ROC-curve) is used as a parameter to demonstrate how well a model performs. While discriminating between signal and background, the signal efficiency is the proportion of the correctly categorized signal events ($t\bar{t}\gamma$) and the background rejection is the correctly assigned fraction of background events ($t\bar{t}$). There are two types of errors during the distinction process:

- a signal event is categorized as background, which is called false negative

- a background event is assigned to the signal category, called false positive

Examples for different types of ROC-curves are displayed in Figure 30. The green curve indicate a good discriminatory power. The orange curve shows that the model assigns the events the wrong way around. If the ROC-curve follows the identity line, then the model doesn't have any discriminatory power.

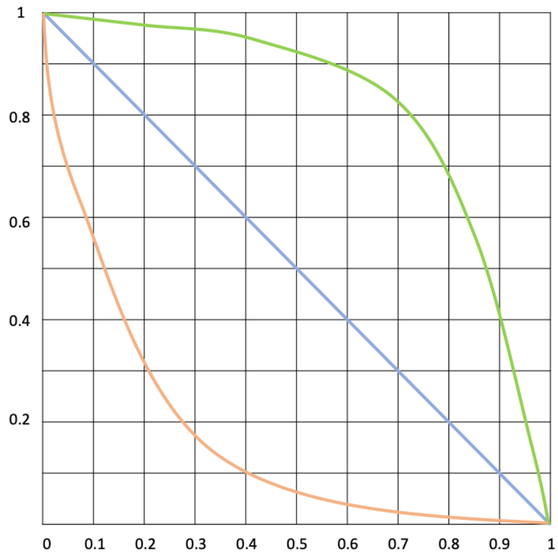


Figure 30: ROC-curve examples

5.4. Results

The first selection category were put into the models and are seen in Figure 31 - 33. The ROC-curves indicate a slight improvement in the split between $t\bar{t}\gamma$- and $t\bar{t}$-events.

In the second selection category the variable PhotonGood0_mvaID was added (Figure 34 - 36). The result has improved, the AUC increased significantly.

Now adding the photon variables for the third selection shows a better success in the distinction process. They can be seen in Figure 37 - 39.

Between the three configuration combinations, there is only a slight difference for the MLP model, but for the BDT model there is a significant improvement in configuration 2 visible. Training- and testresults are nearly identical, so there are no signs of overfitting. The MLP model perfomed better than the BDT model for all selection categories. The third set of variables show the best result.

For all selection categories

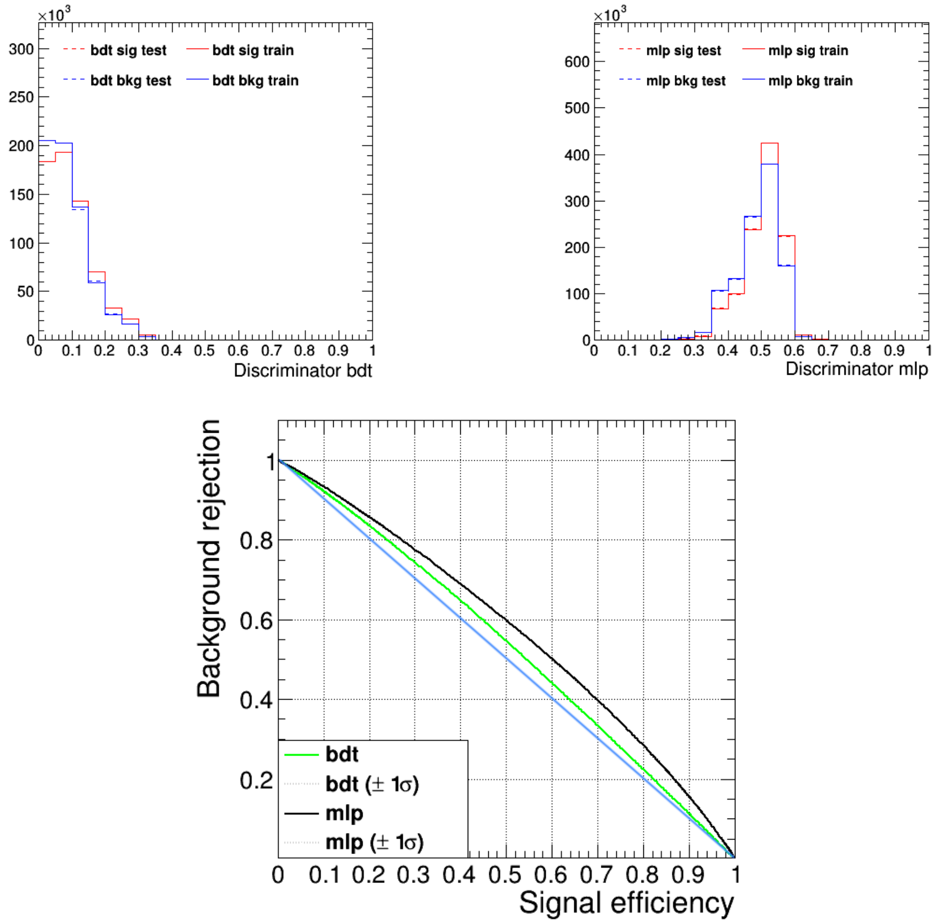


Figure 31: BDT-, MLP-distribution and ROC curve for variables from selection 1 with configuration 1

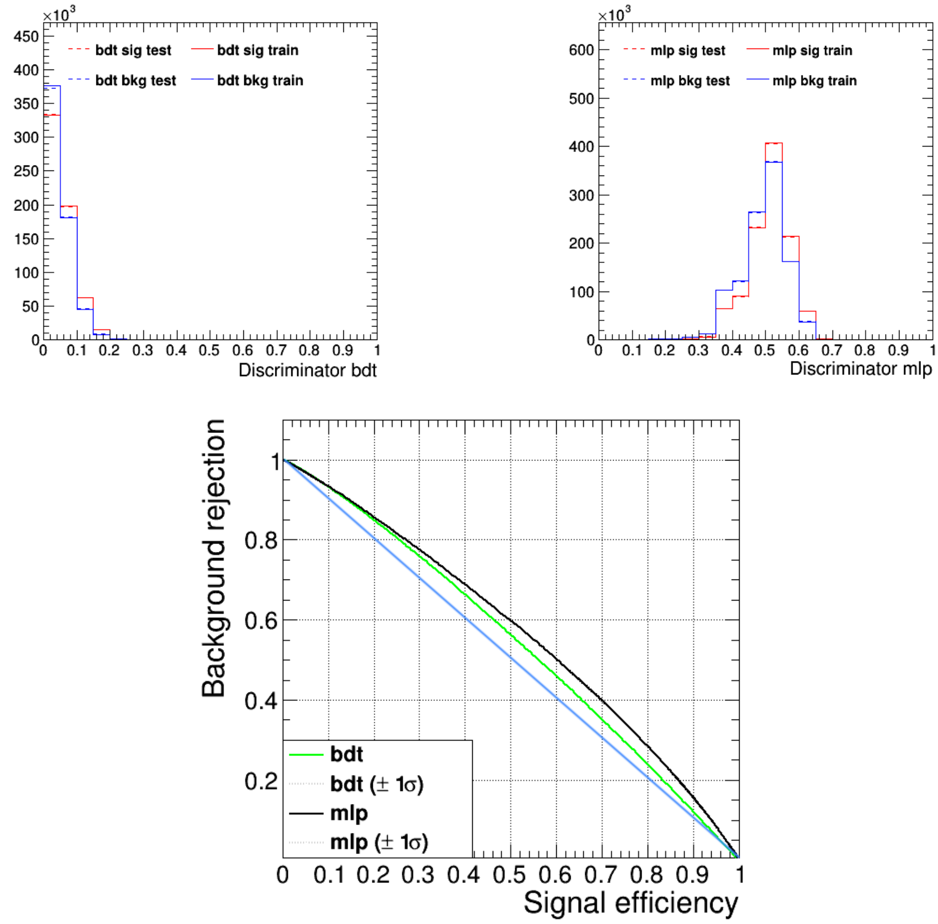


Figure 32: BDT-, MLP-distribution and ROC curve for variables from selection 1 with configuration 2

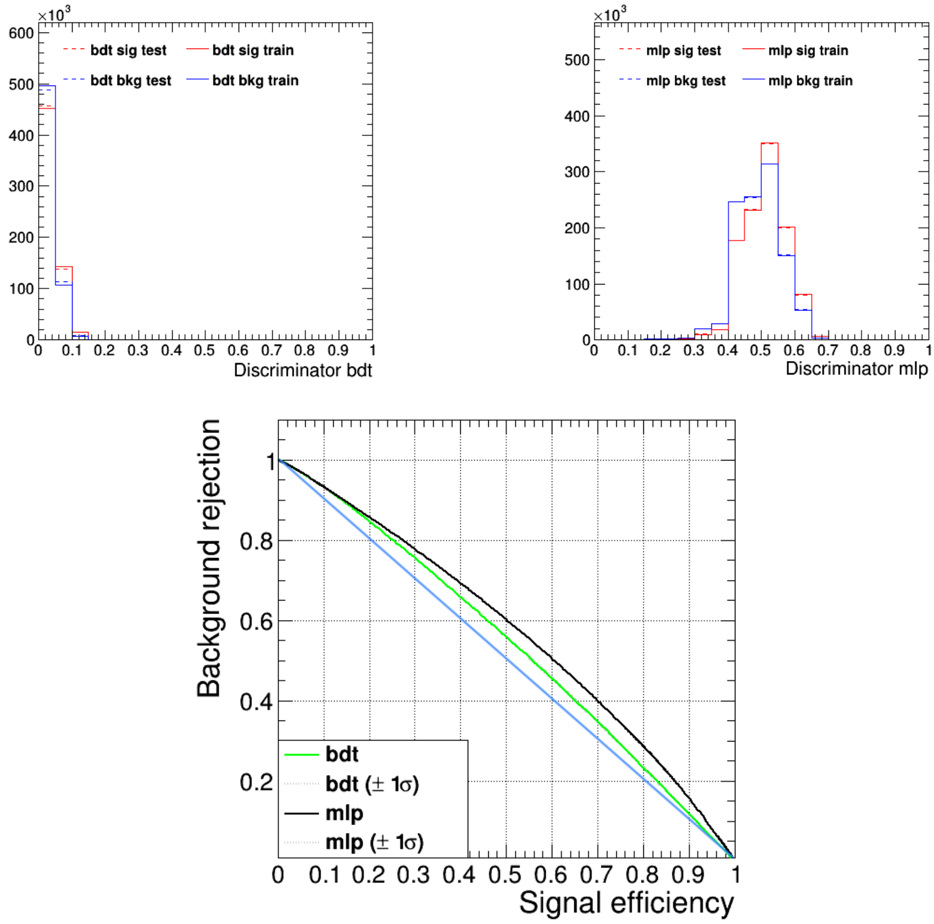


Figure 33: BDT-, MLP-distribution and ROC curve for variables from selection 1 with configuration 3

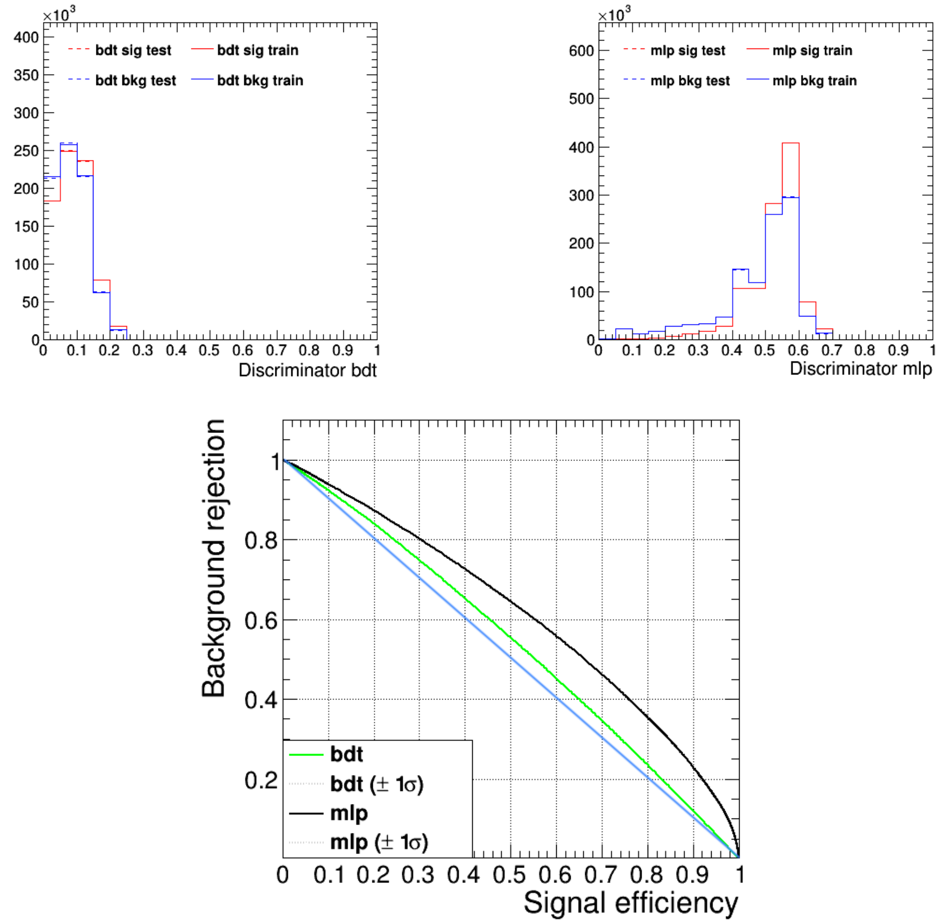


Figure 34: BDT-, MLP-distribution and ROC curve for variables from selection 2 with configuration 1

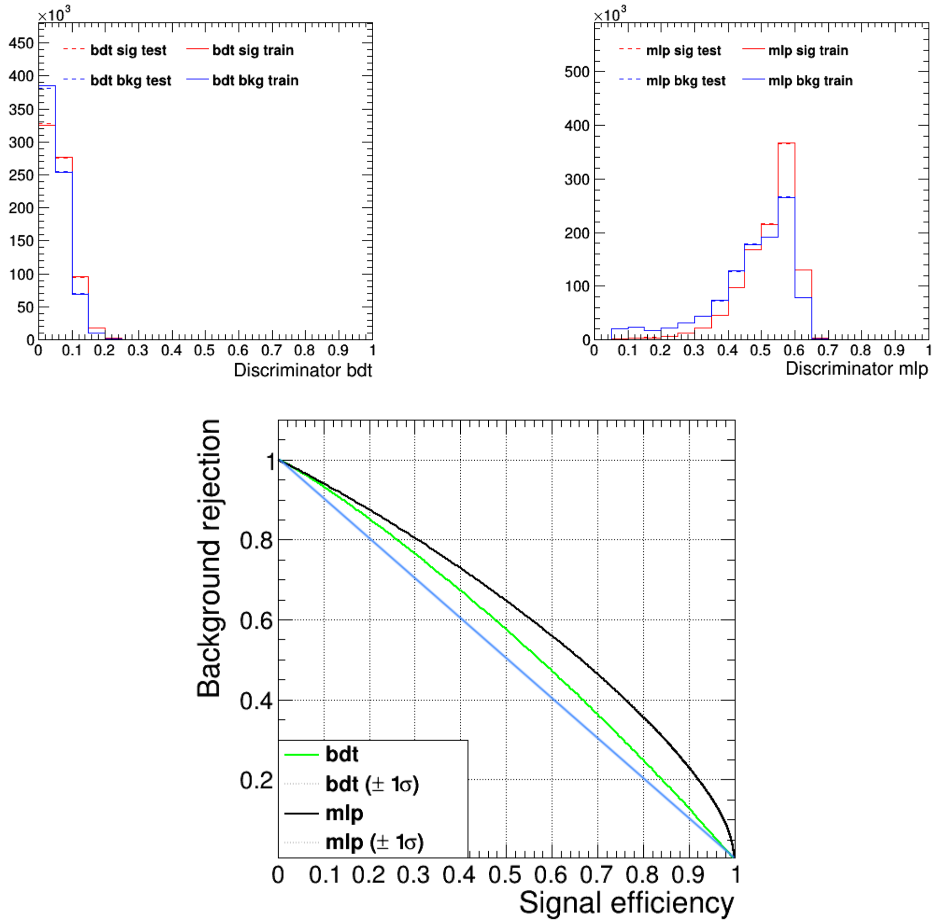


Figure 35: BDT-, MLP-distribution and ROC curve for variables from selection 2 with configuration 2

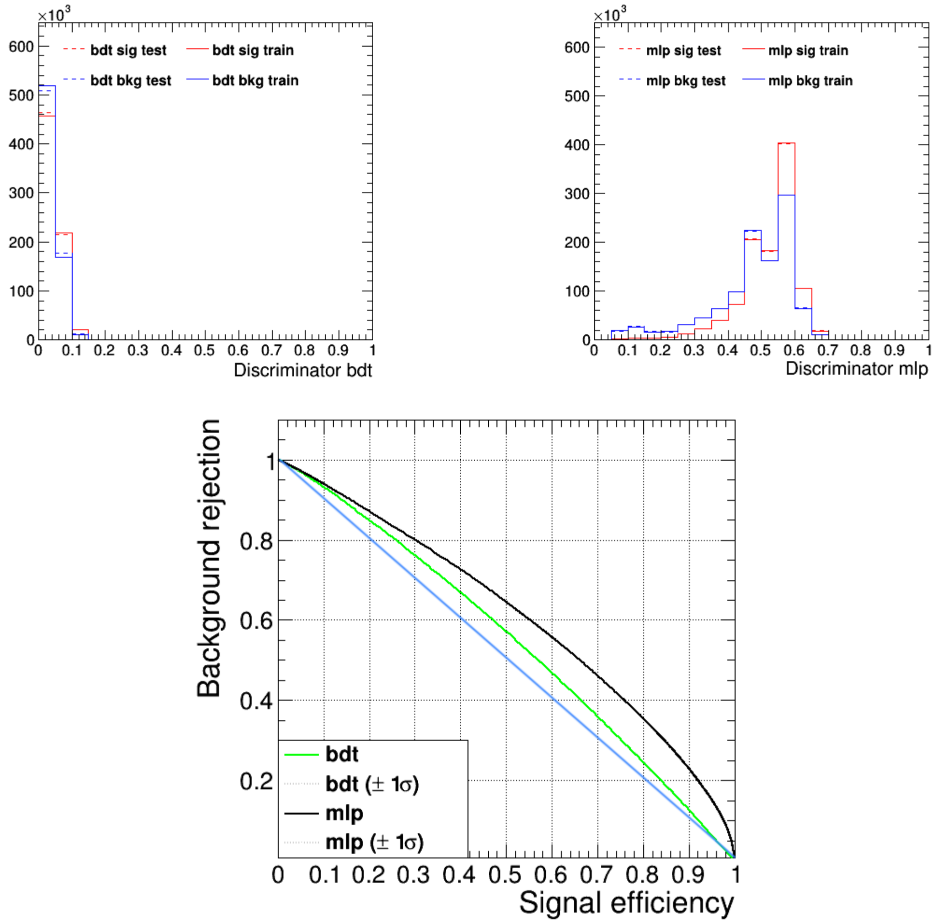


Figure 36: BDT-, MLP-distribution and ROC curve for variables from selection 2 with configuration 3

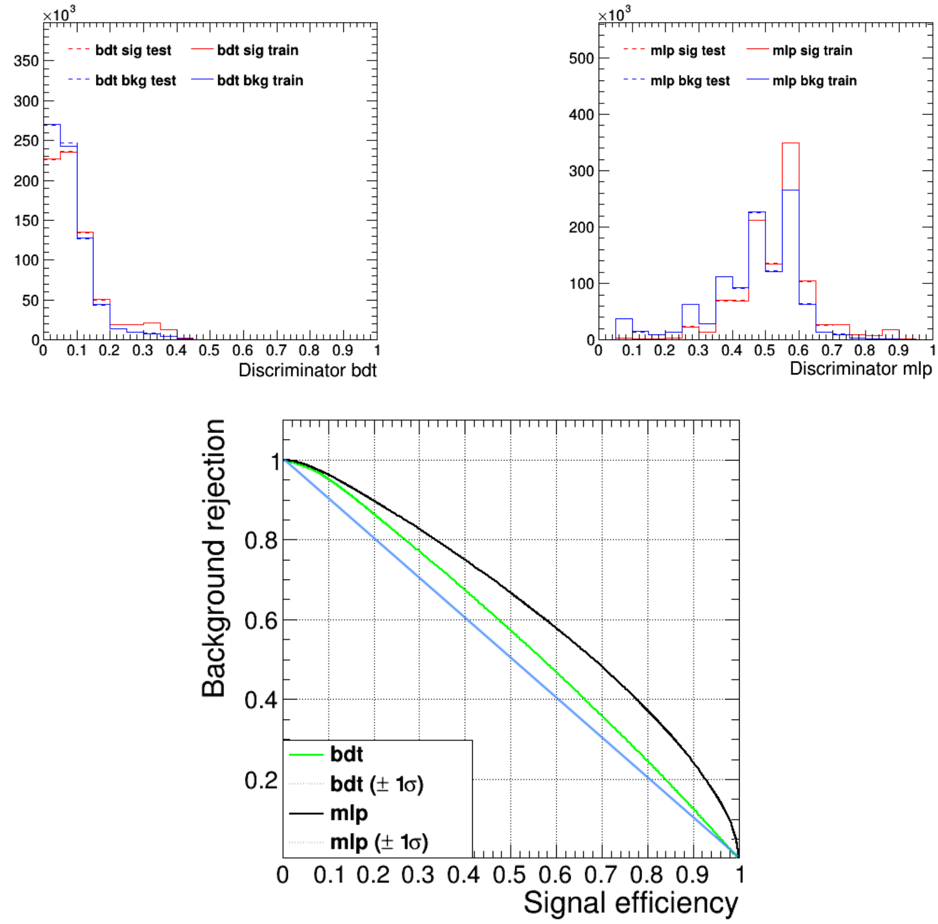


Figure 37: BDT-, MLP-distribution and ROC curve for variables from selection 3 with configuration 1

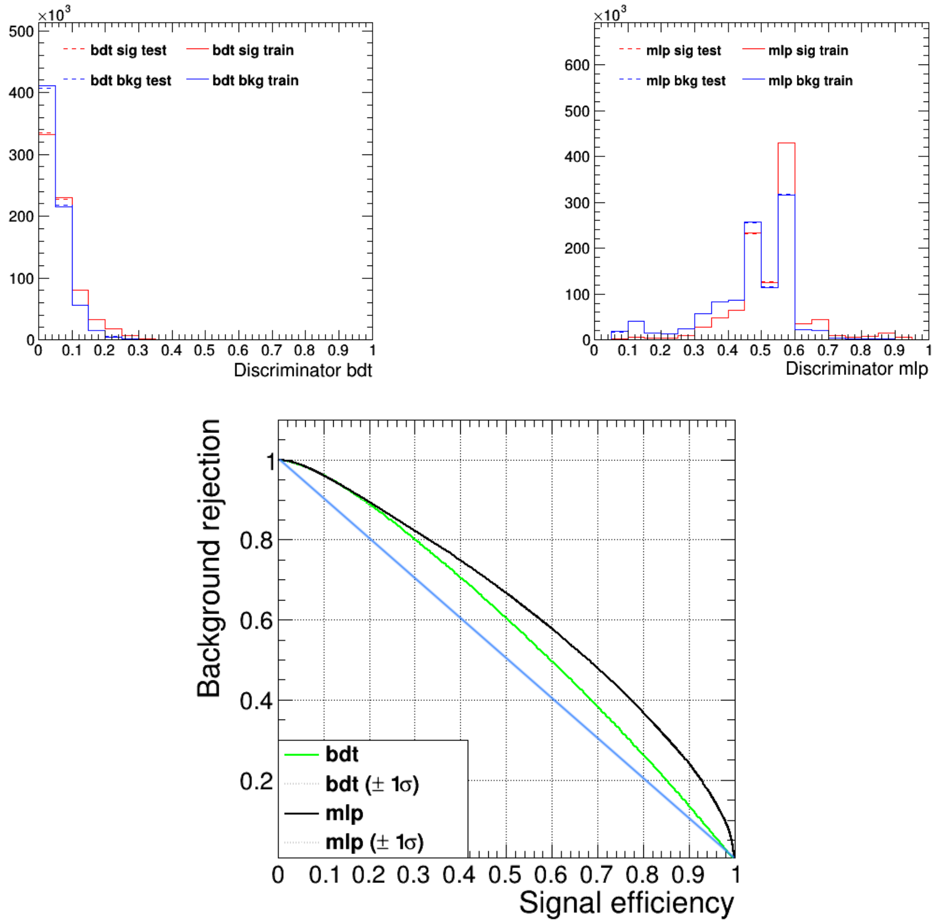


Figure 38: BDT-, MLP-distribution and ROC curve for variables from selection 3 with configuration 2

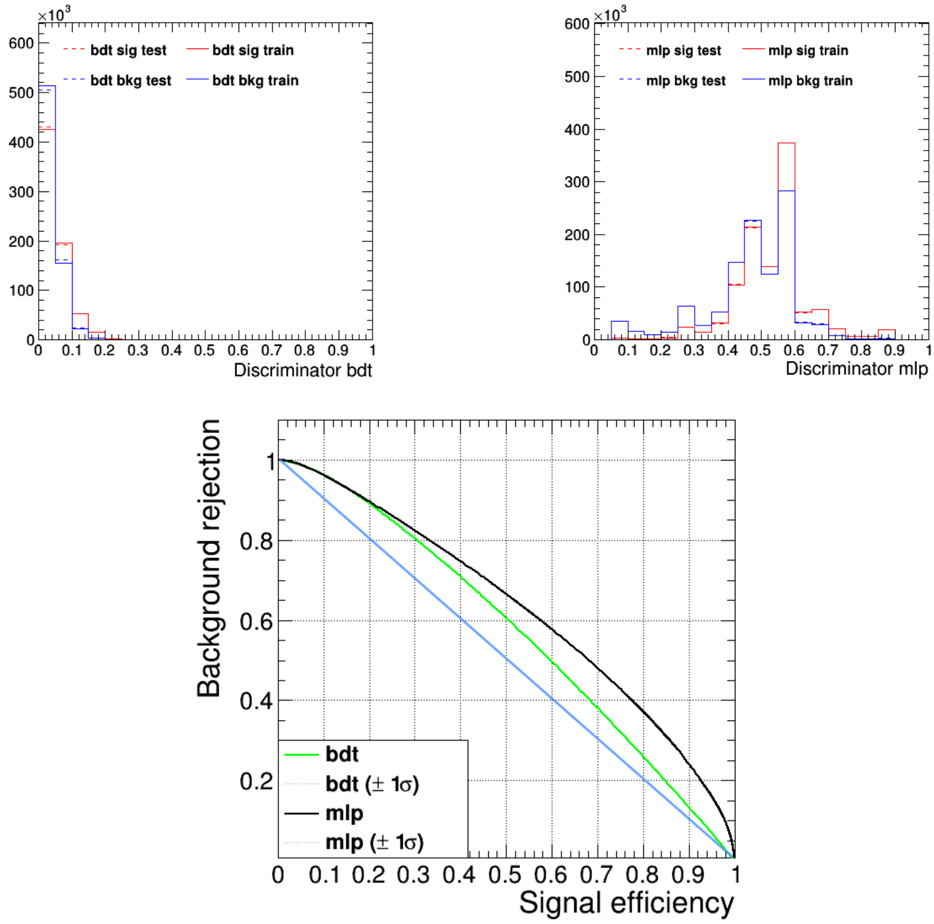


Figure 39: BDT-, MLP-distribution and ROC curve for variables from selection 3 with configuration 3

6. Conclusion

It is very difficult to differentiate between $t\bar{t}\gamma$ and $t\bar{t}$ events because they show very similiar behaviour in the signal region. Nevertheless, a slight improvement was possible by training a MVA with relevant variables. Especially photon variables provide a bigger improvement. MLPs show a better performance than BDTs, which suggests the selection of this type of model.

A. Appendix

A.1. Full list of variables with almost identical distributions

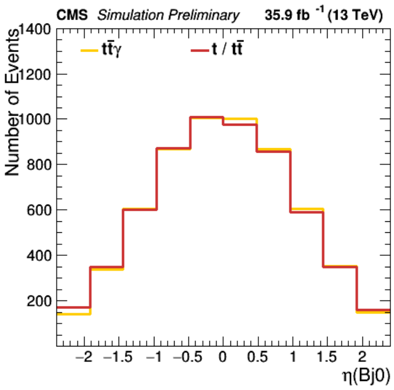


Figure 40: Bj0_eta

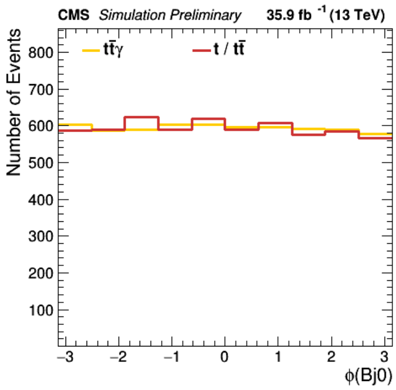


Figure 41: Bj0_phi

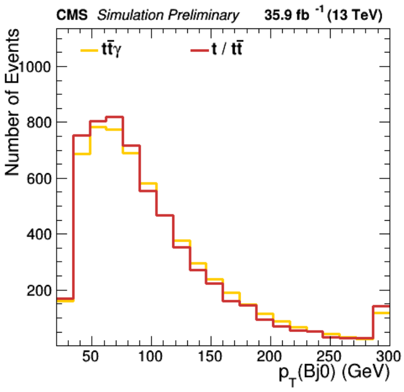


Figure 42: Bj0_pt

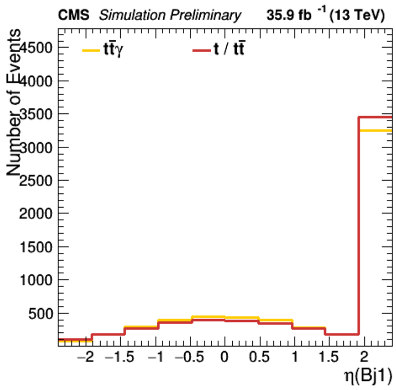


Figure 43: Bj1_eta

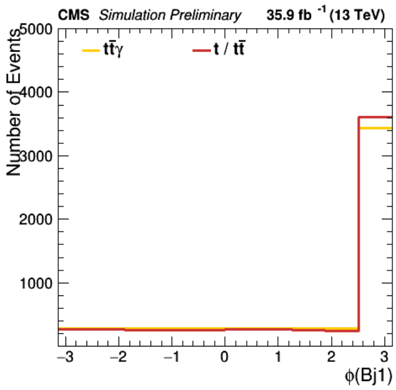


Figure 44: Bj1_phi

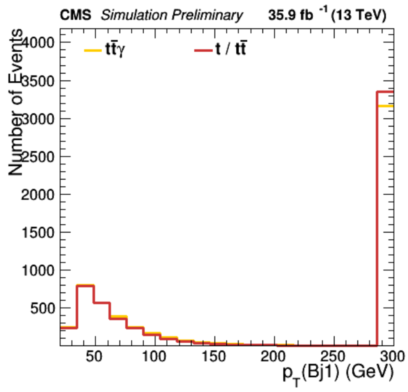


Figure 45: Bj1_pt

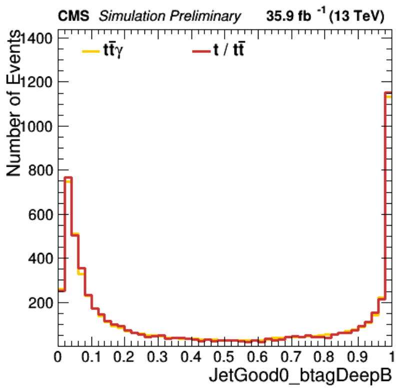


Figure 46: JetGood0_btagDeepB

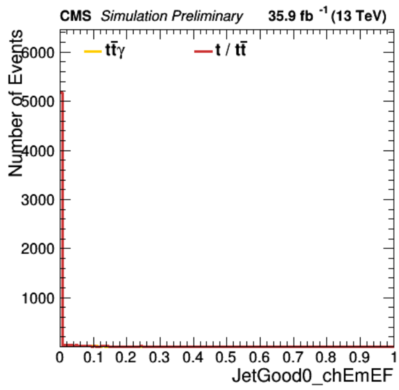


Figure 47: JetGood0_chEmEF

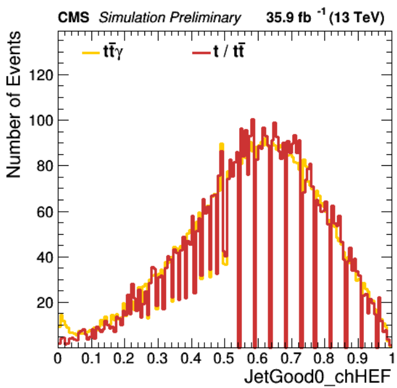


Figure 48: JetGood0_chHEF

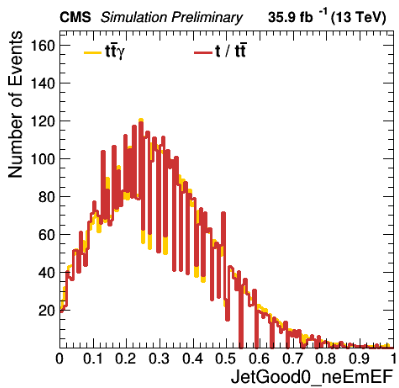


Figure 49: JetGood0_neEmEF

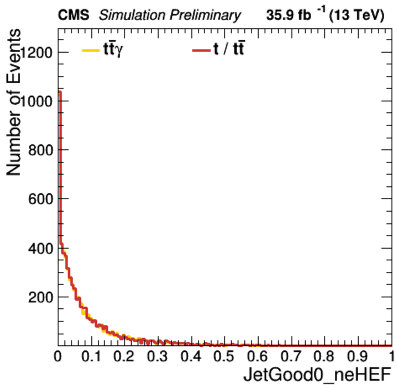


Figure 50: JetGood0_neHEF

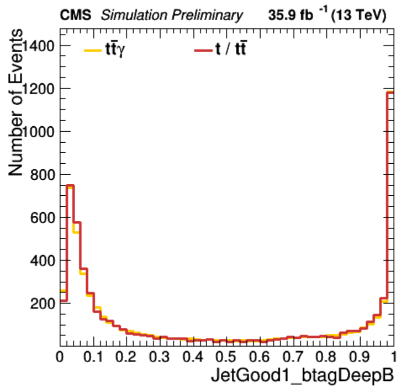


Figure 51: JetGood1_btagDeepB

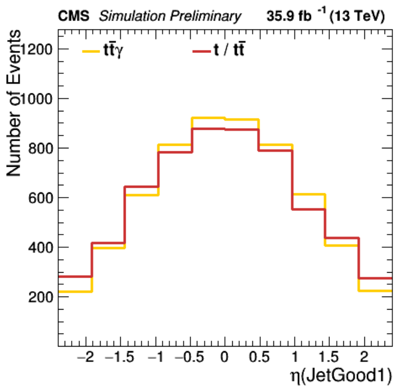


Figure 52: JetGood1_eta

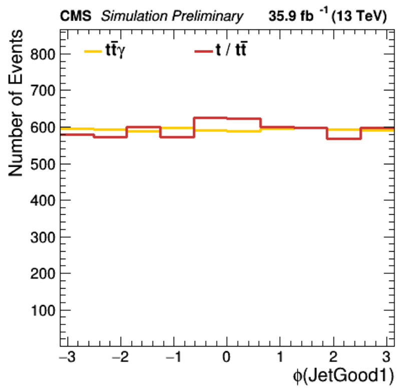


Figure 53: JetGood1_phi

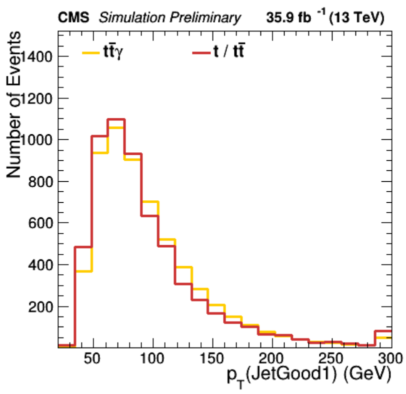


Figure 54: JetGood1_pt

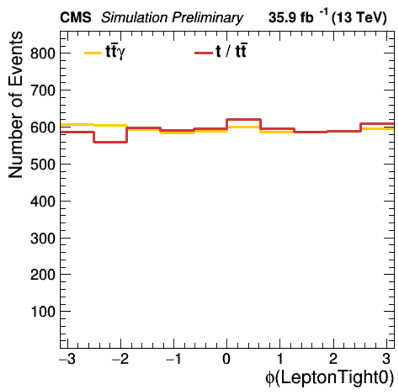


Figure 55: LeptonTight0_phi

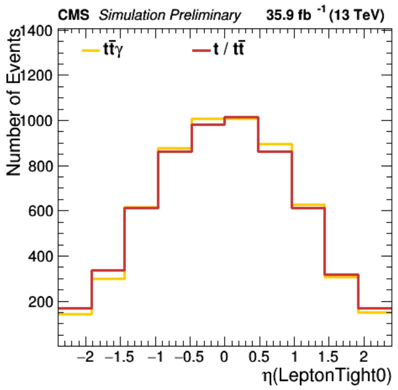


Figure 56: LeptonTight0_eta

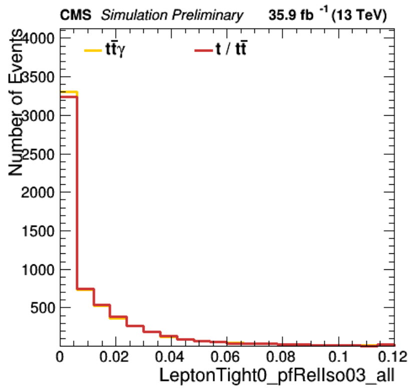


Figure 57: LeptonTight0_pfRelIso03_all

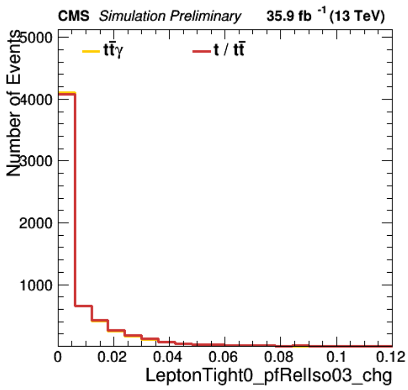


Figure 58: LeptonTight0_pfRelIso03_chg

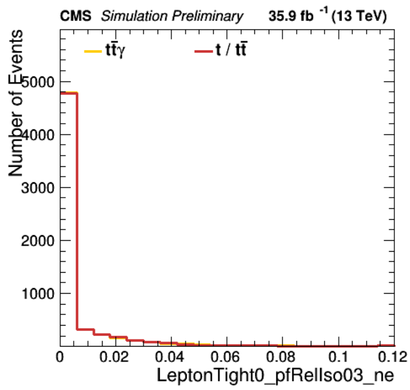


Figure 59: LeptonTight0_pfRelIso03_ne

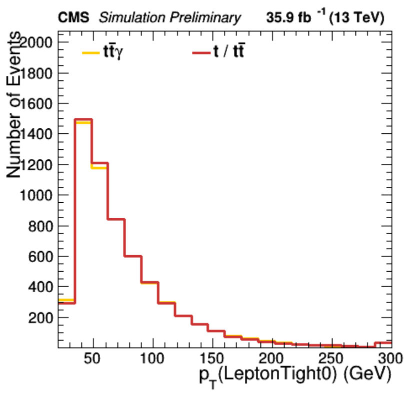


Figure 60: LeptonTight0_pt

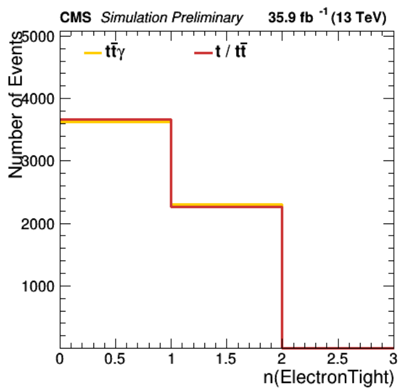


Figure 61: nElectronTight

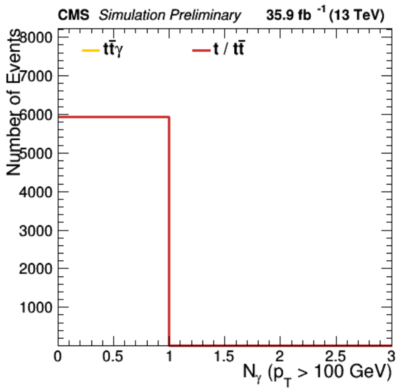


Figure 62: nHighPTPhotons

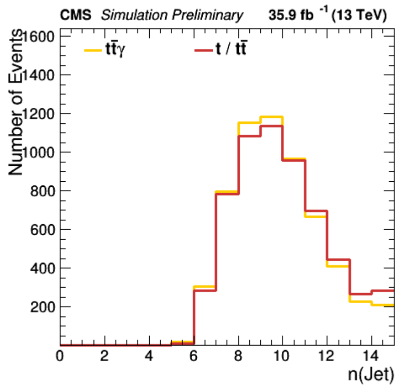


Figure 63: nJet

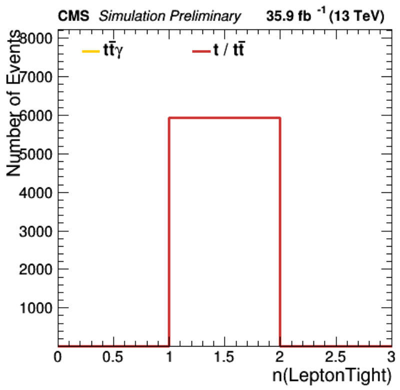


Figure 64: nLeptonTight

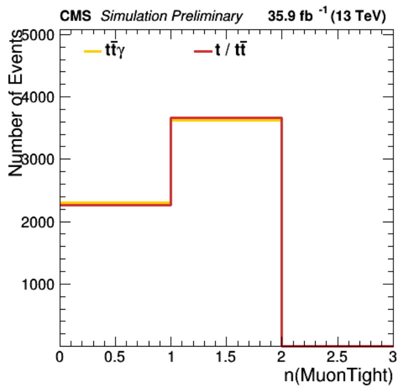


Figure 65: nMuonTight

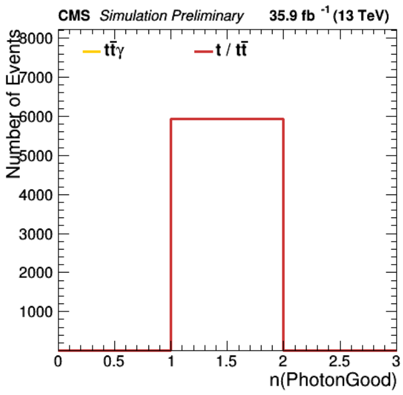


Figure 66: nPhotonGood

A.2. Configuration options for MVA method: BDT

Note: Table taken from [6], p. 116-118.

Option	Default	Description
NTrees	800	Number of trees in the forest
MaxDepth	3	Max depth of the decision tree allowed
MinNodeSize	5%	Minimum percentage of training events required in a leaf node (default: Classification: 5%, Regression: 0.2%)
nCuts	20	Number of grid points in variable range used in finding optimal cut in node splitting
BoostType	AdaBoost	Boosting type for the trees in the forest (note: AdaCost is still experimental) (AdaBoost, RealAdaBoost, Bagging, AdaBoostR2, Grad)
AdaBoostR2Loss	Quadratic	Type of Loss function in AdaBoostR2 (Linear, Quadratic, Exponential)
UseBaggedGrad	False	Use only a random subsample of all events for growing the trees in each iteration. (Only valid for GradBoost)
Shrinkage	1	Learning rate for GradBoost algorithm
AdaBoostBeta	0.5	Learning rate for AdaBoost algorithm
UseRandomisedTrees	False	Determine at each node splitting the cut variable only as the best out of a random subset of variables (like in RandomForests)
UseNvars	2	Size of the subset of variables used with RandomisedTree option
UsePoissonNvars	True	Interpret UseNvars not as fixed number but as mean of a Possion distribution in each split with RandomisedTree option
BaggedSampleFraction	0.6	Relative size of bagged event sample to original size of the data sample (used whenever bagging is used (i.e. UseBaggedGrad, Bagging, ..))
UseYesNoLeaf	True	Use Sig or Bkg categories, or the purity=S/(S+B) as classification of the leaf node ->Real-AdaBoost

NegWeightTreatment	InverseBoost-NegWeights	How to treat events with negative weights in the BDT training (particular the boosting) : IgnoreInTraining; Boost With inverse boostweight (InverseBoost-NegWeights); Pair events with negative and positive weights in traning sample and *annihilate* them (PairNegWeights-Global, experimental!), Pray
NodePurityLimit	0.5	In boosting/pruning, nodes with purity >NodePurityLimit are signal; background otherwise.
SeparationType	GiniIndex	Separation criterion for node splitting (CrossEntropy, GiniIndex, GiniIndexWithLaplace, MisClassificationError, SdivSqrtSPlusB, RegressionVariance)
DoBoostMonitor	False	Create control plot with ROC integral vs tree number
UseFisherCuts	False	Use multivariate splits using the Fisher criterion
MinLinCorrForFisher	0.8	The minimum linear correlation between two variables demanded for use in Fisher criterion in node splitting
UseExclusiveVars	False	Variables already used in fisher criterion are not anymore analysed individually for node splitting
DoPreselection	False	Apply automatic pre-selection for 100% efficient signal (bkg) cuts prior to training
RenormByClass	False	Individually re-normalize each event class to the original size after boosting
SigToBkgFraction	1	Sig to Bkg ratio used in Training (similar to NodePurityLimit, which cannot be used in real adaboost)
PruneMethod	NoPruning	Note: for BDTs use small trees (e.g.MaxDepth=3) and NoPruning; Pruning: Method used for pruning (removal) of statistically insignificant branches (NoPruning, ExpectedError, CostComplexity)
PruneStrength	0	Pruning strength
PruningValFraction	0.5	Fraction of events to use for optimizing automatic pruning.

nEventsMin	0	deprecated: Use MinNodeSize (in % of training events) instead
GradBaggingFraction	0.6	deprecated: Use *BaggedSampleFraction* instead: Defines the fraction of events to be used in each iteration, e.g. when UseBaggedGrad=kTRUE.
UseNTrainEvents	0	deprecated: Use *BaggedSampleFraction* instead: Number of randomly picked training events used in randomised (and bagged) trees
NNodesMax	0	deprecated: Use MaxDepth instead to limit the tree size

Table 2: Full list of configuration options and their default values (BDT)

A.3. Configuration options for MVA method: MLP

Note: Table taken from [6], p. 98-99.

Option	Default	Description
NCycles	500	Number of training cycles
HiddenLayers	N,N-1	Specification of hidden layer architecture
NeuronType	sigmoid	Neuron activation function type
RandomSeed	1	Random seed for initial synapse weights (0 means unique seed for each run; default value '1')
EstimatorType	MSE	MSE (Mean Square Estimator) for Gaussian Likelihood or CE (Cross- Entropy) for Bernoulli Likelihood, linear, sigmoid, tanh, radial
NeuronInputType	sum	Neuron input function type (sum, sqsum, abssum)
TrainingMethod	BP	Train with Back-Propagation (BP), BFGS Algorithm (BFGS), or Genetic Algorithm (GA - slower and worse)
LearningRate	0.02	ANN learning rate parameter
DecayRate	0.01	Decay rate for learning parameter
TestRate	10	Test for overtraining performed at each #th epochs
EpochMonitoring	False	Provide epoch-wise monitoring plots according to TestRate (caution: causes big ROOT output file!)

Sampling	1	Only 'Sampling' (randomly selected) events are trained each epoch
SamplingEpoch	1	Sampling is used for the first 'SamplingEpoch' epochs, afterwards, all events are taken for training
SamplingImportance	1	The sampling weights of events in epochs which successful (worse estimator than before) are multiplied with SamplingImportance, else they are divided.
SamplingTraining	True	The training sample is sampled
SamplingTesting	False	The testing sample is sampled
ResetStep	50	How often BFGS should reset history
Tau	3	LineSearch size step
BPMode	sequential	Back-propagation learning mode: sequential or batch
BatchSize	-1	Batch size: number of events/batch, only set if in Batch Mode, -1 for BatchSize=number_of_events
ConvergenceImprove	1e-30	Minimum improvement which counts as improvement (<0 means automatic convergence check is turned off)
ConvergenceTests	-1	Number of steps (without improvement) required for convergence (<0 means automatic convergence check is turned off)
UseRegulator	False	Use regulator to avoid over-training
UpdateLimit	10000	Maximum times of regulator update
CalculateErrors	False	Calculates inverse Hessian matrix at the end of the training to be able to calculate the uncertainties of an MVA value
WeightRange	1	Take the events for the estimator calculations from small deviations from the desired value to large deviations only over the weight range

Table 3: Full list of configuration options and their default values (MLP)

B. References

[1] M. Aaboud, G. Aad, B. Abbott, D. C. Abbott, O. Abdinov, B. Abeloos, D. K. Abhayasinghe, S. H. Abidi, O. S. AbouZeid, et al. *Measurements of inclusive and differential fiducial cross-sections of $t\bar{t}\gamma$ production in leptonic final states at $\sqrt{s} = 13$ TeV in ATLAS*. The European Physical Journal C, 79(5) (2019). ISSN 1434-6052. doi:10.1140/epjc/s10052-019-6849-6. URL http://dx.doi.org/10.1140/epjc/s10052-019-6849-6.

[2] J. Erdmann. *Measurement of the inclusive ttgamma cross section at sqrt(s) = 7 TeV with the ATLAS detector* (2012). 1206.5696.

[3] Y. Coadou. *Boosted Decision Trees and Applications*. EPJ Web of Conferences, 55:02004– (2013). doi:10.1051/epjconf/20135502004.

[4] J. Nazzal, I. El-Emary, S. Najim. *Multilayer Perceptron Neural Network (MLPs) For Analyzing the Properties of Jordan Oil Shale*. World Applied Sciences Journal, 5 (2008).

[5] P. Marius, V. Balas, L. Perescu-Popescu, N. Mastorakis. *Multilayer perceptron and neural networks*. WSEAS Transactions on Circuits and Systems, 8 (2009).

[6] A. Hoecker, P. Speckmayer, J. Stelzer, J. Therhaag, E. von Toerne, H. Voss, M. Backes, T. Carli, O. Cohen, A. Christov, S. Henrot-Versillé, M. Jachowski, K. Kraszewski, Y. Mahalalel, R. Ospanov, X. Prudent, D. Schouten, F. Tegenfeldt, A. Robert, A. Voigt, K. Voss, M. Wolter, A. Zemla. *TMVA - Toolkit for Multivariate Data Analysis* (2009). physics/0703039v5.

C. List of Figures

D. List of Tables