Bachelor Thesis

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Optimization of Machine Learning for the Leptoquark Search using CERN ATLAS Data

Janick Böhm

Supervisor: doc. Dr. André Sopczak
Field of study: Electrical Engineering and Computer Science
May 2023

BACHELOR‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Böhm Janick**	Personal ID number:	**498014**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Cybernetics**		
Study program:	**Electrical Engineering and Computer Science**		

II. Bachelor’s thesis details

Bachelor’s thesis title in English:

Optimization of Machine Learning for the Leptoquark Search Using CERN ATLAS Data

Bachelor’s thesis title in Czech:

Optimalizace vyhledávání leptoquarků pomocí strojového učení v datech z CERN ATLAS experiment

Guidelines:

At the Large Hadron Collider (LHC) at CERN protons are collided and the collisions are recorded by the ATLAS detector. A motivation to construct the LHC has been the search for new particles. Such new particles could be Lep-toquarks. Leptoquarks are predicted by several theories, but so far have not been detected in the recorded data. Their mass is unknown. Leptoquark signal events have been simulated, together with other events resulting from back-ground reactions. The task is to recognize these simulated signal events automatically from other (background) events using the techniques of machine learning and possibly deep learning.
Instructions:
1. Get familiar with the data and basic principles of searching for elementary particles in high-energy physics.
2. Get familiar with the existing implementation of classifiers to separate events of interest from the background, using high-level features and classical machine-learning techniques.
3. Develop and optimize either classical machine learning or deep-learning algorithms based on high-level features, taking into account statistical and systematic uncertainties of the feature simulations.
4. Evaluate its performance on simulated data and determine the leptoquark sensitivity over a large mass range.
5. Compare and discuss the obtain sensitivity with previous results.

Bibliography / sources:

[1] https://atlas.cern (“learn more”) An introduction to the ATLAS experiment for the public
[2] Dan Guest et al. Deep Learning and its application to LHC Physics. Annu. Rev. Nucl. Part. Sci. 2018, 68:1-22
[3] Pierre Baldil et al: Searching for Exotic Particles in High-Energy Physics. arXiv:1402.4735
[4] ATLAS Collaboration: Search for pair production of third-generation scalar leptoquarks decaying into a top quark and a tau-lepton in pp collisions at $\sqrt{s}$ = 13 TeV with the ATLAS detector. JHEP 06 (2021) 179
[5] A.Sopczak, Searches for Leptoquarks with the ATLAS Detector, arXiv:2107.10094
[6] R.Duda, P.Hart, D.Stork: Pattern classification. Willey-Interscience, 2000
[7] Goodfellow, Bengio, Courville: Deep learning. MIT Press. 2016
[8] Lukas Vicenik, Machine Learning for the Leptoquark Search Using CERN ATLAS Data, 14 Jun 2022, https://cds.cern.ch/record/2812370

Name and workplace of bachelor’s thesis supervisor:

doc. Dr. André Sopczak   High Energy Physics, IEAP CTU in Prague

Name and workplace of second bachelor’s thesis supervisor or consultant:

Date of bachelor’s thesis assignment: **17.10.2022**   Deadline for bachelor thesis submission: **26.05.2023**

Assignment valid until: **22.09.2024**

doc. Dr. André Sopczak Supervisor’s signature	doc. Ing. Zdeněk Müller, Ph.D. Head of department’s signature	prof. Mgr. Petr Páta, Ph.D. Dean’s signature

III. Assignment receipt

The student acknowledges that the bachelor’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor’s thesis, the author must state the names of consultants and include a list of references.

.

Date of assignment receipt   Student’s signature

Acknowledgements

I would like to express my gratitude to my supervisor, doc. Dr. André Sopczak, for his patience, support and guidance over the course of the thesis. I would also like to thank my family and my better half for their encouragement and words of assurance over the years of my studies, and the years to come.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis. I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60 (1) of the Act.

Prague, May 26, 2023

Abstract

The Leptoquark is among the undiscovered particles which are being searched for in the Large Hadron Collider. Monte Carlo simulated events of proton-to-proton collisions corresponding to the Leptoquark are studied with the ATLAS detector. The luminosity of the produced samples corresponds to the recorded data of 140 fb^{-1}.

Four machine learning algorithms are used (TabNet, XGBoost, MLP, and Bayesian MLP) to train models to separate events on the 2lSS + 1τ channel belonging to the pair-production mode of Leptoquark from various background processes, including $\mathrm{t\bar{t}H}$, $\mathrm{t\bar{t}W}$, $\mathrm{t\bar{t}Z}$, $\mathrm{t\bar{t}}$, VV and other minor processes.

The feature importance of the top performing models is constructed and utilized to produce more efficient models with improved sensitivity. In addition, the expected upper limit of cross-section for the pair-production of Leptoquark at 95% confidence level is calculated and compared to existing results.

Keywords: Leptoquark, machine learning, neural networks, classification, cross-section, particle physics, TabNet, XGBoost, ATLAS, CERN, ROOT

Supervisor: doc. Dr. André Sopczak
Husova 240/5, 11000 Prague 1

Abstrakt

Leptokvark patří mezi dosud neobjevené částice, které se hledají ve Velkém hadronovém urychlovači. Detektorem ATLAS jsou studovány Monte Carlo simulované události srážek protonu s protonem odpovídající režimu párové produkce leptokvarku. Luminozita vytvořených vzorků odpovídá 140 fb^{-1}.

K trénování modelů pro oddělení událostí v kanálu 2lSS + 1τ patřících do režimu párové produkce leptokvarku od různých procesů na pozadí, včetně $t\bar{t}H$, $t\bar{t}W$, $t\bar{t}Z$, $t\bar{t}$, VV a dalších vedlejších procesů, se používají čtyři algoritmy strojového učení (TabNet, XGBoost, MLP a Bayesův MLP).

Důležitost příznaků nejvýkonnějších modelů je konstruována a využita k vytvoření efektivnějších modelů se zvýšenou citlivostí. Kromě toho je vypočtena očekávaná horní mez cross-section pro párovou produkci leptokvarku na 95% CL a porovnána s dosavadními výsledky.

Klíčová slova: Leptokvark, strojové učení, neuronové sítě, klasifikace, cross-section, částicová fyzika, TabNet, XGBoost, ATLAS, CERN, ROOT

Překlad názvu: Optimalizace vyhledávání leptoquarků pomocí strojového učení v datech z CERN ATLAS

Contents

Figures 1

Tables 5

Introduction 7

Part I
Theory & Outline

1 State of the Art Research 11

2 Theoretical Background 13
2.1 Cross-section . . . 13
2.2 Luminosity . . . 13

3 The Standard Model 15
3.1 Quarks . . . 16
3.2 Leptons . . . 16

4 Beyond the Standard Model 17
4.1 Leptoquarks . . . 17

5 Experimental and Simulated Data 19
5.1 Monte Carlo Simulation . . . 19
5.2 Processes . . . 20
5.3 Event Selection . . . 20
5.4 Event Weights . . . 22
5.5 Process Yields . . . 23

6 Frameworks 27
6.1 ROOT Framework . . . 27
6.1.1 Uproot . . . 27
6.2 Python . . . 28
6.3 Optuna . . . 28
6.4 TRExFitter . . . 28

7 Machine Learning 31
7.1 Neural Network . . . 31
7.1.1 Artificial Neuron . . . 31
7.1.2 Multilayer Perceptron . . . 33
7.1.3 Loss Function . . . 34
7.2 Bayesian Neural Network . . . 34
7.3 Decision Trees . . . 35
7.3.1 Ensemble Methods . . . 35
7.3.2 Bagging . . . 36
7.3.3 Boosting . . . 36
7.4 XGBoost . . . 37
7.4.1 XGBoost Hyperparameters . . . 38
7.5 TabNet . . . 38
7.5.1 TabNet Hyperparameters . . . 40
7.6 Performance Metrics . . . 41
7.6.1 Confusion Matrix . . . 41
7.7 Significance . . . 44
7.7.1 Confidence Level . . . 44

Part II
Implementation

8 Data Processing 47
8.1 Data Selection . . . 47
8.2 Training & Testing Datasets . . . 48

9 Selection of Best Models 49
9.1 Code . . . 49
9.2 Choice of Mass for Testing . . . 49
9.3 Neural Network . . . 50
9.4 TabNet . . . 51
9.4.1 Evaluated TabNet Models . . . 54
9.5 XGBoost . . . 54
9.5.1 Evaluated XGBoost Models . . . 57
9.6 Selected Models . . . 58
9.7 Optimal Threshold . . . 59

10 Feature Importance 63
10.1 Feature Ranking . . . 63
10.2 Feature Correlation . . . 66

11 Efficient Models 67
11.1 Training on Events Without Preselection . . . 68
11.1.1 Improved Models . . . 68
11.1.2 Network Output . . . 71

12 Binary Classifier 73

13 Results 75
13.1 Expected Limits . . . 75
13.1.1 Employment of TRExFitter 75
13.2 Comparison of Expected Upper Limit of Cross-section . . . 78

14 Conclusion 81

Bibliography 83

Appendices

A Pre-selection 89

B Python Libraries 91

C Confusion Matrices 93

D Feature Importance 95

E Feature Comparison of Different Masses 99

F Network Output 101

G Binary Classifier Results 107

H TRExFitter Config for Cross-section Estimate 109

I Expected Upper Limit 111

Figures

1.1 Diagram of the Large Hadron Collider with the four largest detectors [1]. . . . 11
1.2 Cross-section diagram of the ATLAS detector [2]. . . . 12

3.1 Standard Model of Elementary Particles [3]. . . . 15

4.1 Feynman diagrams for Leptoquark pair-production (left), single-production (centre), off-shell production (right) [4]. . . . 17
4.2 Feynman diagrams for the same final state for pair-production mode of Leptoquark (left) and production Higgs boson ($t\bar{t}H(H \to \tau\tau)$) (right) [4]. . . . 18

7.1 Role of a single neuron [5]. . . . 32
7.2 Structure of a neural network [6]. . . . 33
7.3 Structure of a decision tree [7]. . . . 35
7.4 Computation of the output [8]. . . . 40
7.5 Confusion matrix for binary classification [9]. . . . 42
7.6 ROC curve with additional annotations [10]. . . . 43

9.1 Optuna coordinate plot for TabNet trained on all LQ masses and tested on mass of 700 GeV, with LQ $\sigma = 1\,\mathrm{pb}$. . . . 52
9.2 Optuna hyperparameter importance plot for TabNet trained on all LQ masses and tested on mass of 700 GeV, with LQ $\sigma = 1\,\mathrm{pb}$. . . . 53
9.3 Optuna coordinate plot for XGBoost trained on all LQ masses and tested on mass of 700 GeV, with LQ $\sigma = 1\,\mathrm{pb}$. . . . 55
9.4 Optuna hyperparameter importance plot for XGBoost trained on all LQ masses and tested on mass of 700 GeV, with LQ $\sigma = 1\,\mathrm{pb}$. . . . 56
9.5 Network output corresponding to the signal class before and after threshold cut for the selected TabNet model tested on mass of 500 GeV, with LQ $\sigma = 0.01\,\mathrm{pb}$. . . . 59
9.6 Significance approximation per threshold and signal to background ratio for selected TabNet model tested on mass of 500 GeV, with LQ $\sigma = 0.01\,\mathrm{pb}$. . . . 60

9.7 Weighted ROC curves for the selected TabNet model (upper) and XGBoost model (lower), for combined masses of LQ, with LQ $\sigma = 0.001\,\text{pb}$. 61

10.1 Feature importance of top 20 features of the selected TabNet model trained on all features, using all LQ masses, computed as averages of 20 trials. 63
10.2 Feature importance of top 20 features of the selected XGBoost model trained on all features, using all LQ masses, computed as averages of 20 trials. 64
10.3 Histograms with distributions of the top four features, with LQ $\sigma = 0.001\,\text{pb}$. 65
10.4 Pearson correlation coefficients for the 10 most important features. 66

11.1 Weighted ROC curves of the TabNet model trained on all events without preselection and tested on events with preselection, for combined masses of LQ, with LQ $\sigma = 0.001\,\text{pb}$. 68
11.2 Weighted ROC curves of the XGBoost model trained on all events without preselection and tested on events with preselection, for combined masses of LQ, with LQ $\sigma = 0.001\,\text{pb}$. 69
11.3 Confusion matrices for models trained on preselected events only (upper) and models trained on events without preselection (lower). Tested on all LQ masses in preselection, with LQ $\sigma = 0.001\,\text{pb}$. 70
11.4 Network output corresponding to the signal class for three different test masses (500, 1000, 1500 GeV), with LQ $\sigma = 0.01\,\text{pb}$. 72

12.1 Confusion matrices for XGBoost multi-class classifier (left) and binary classifier (right) trained on preselected events only. Tested on all LQ masses in preselection, with LQ $\sigma = 0.001\,\text{pb}$. 74

13.1 Comparison of limit obtained by previous result [11] (upper) and limit obtained during by this analysis (lower) using the TabNet model. 79

C.1 Confusion matrix for the TabNet model trained on preselected data without using a threshold (upper), and with (lower). 94

E.1 Histograms with distributions of two features with high importance for 500 GeV (left) and 1600 GeV (right), with LQ $\sigma = 0.01\,\text{pb}$. 99

F.1 Network output corresponding to the signal class for all masses of both TabNet and XGBoost models, with LQ $\sigma = 0.01\,\text{pb}$. 105

G.1 Expected upper limit plots with logarithmic scale, obtained using the XGBoost binary classifier trained on data without preselection. 107

I.1 Expected upper limit plots with logarithmic scale, obtained using the XGBoost model trained on data without preselection. 111
I.2 Expected upper limit plots with logarithmic scale, obtained using the TabNet model trained on data without preselection. 111

I.3 Expected upper limit plots with linear scale, obtained using the TabNet model (upper) and XGBoost model (lower) trained on data without preselection. 112

Tables

5.1 Summary of preselection criteria. A complete definition of the preselection region is given in Appendix A. 20

5.2 Four different types of lepton represented by four possible values of "lep_ID_0" and "lep_ID_1". 21

5.3 Number of raw events and yields for each LQ mass (500 - 1600 GeV), using theoretical cross-sections. 23

5.4 Number of raw events and yield for each process and combination of processes, using theoretical cross-sections. 24

5.5 DSID for each process. LQ DSIDs arranged in order from 500 to 1600 GeV); $t\bar{t}$W DSIDs consist of ttW2210 and ttW2210_EW samples, respectively; "Other" DSIDs consist of threeTop, VH, VVV, fourTop, rareTop, WtZ and tZ samples in this order. 25

5.6 Yield for V6 and V8 dataset for each LQ mass (500 - 1600 GeV), with LQ $\sigma = 1\,\mathrm{pb}$). 25

8.1 Recorded number of raw events and yield before and after events with negative weights were removed, with LQ $\sigma = 1\,\mathrm{pb}$. 47

9.1 Combined weight of events in the test set for each background and signal with mass of 500 GeV, with LQ $\sigma = 0.01\,\mathrm{pb}$. 50

9.2 Significance (simplified) for different variations of MLP and Bayesian MLP models trained on all LQ masses, and tested on 500 GeV. 50

9.3 Significance (simplified) for different variations of TabNet models trained on all LQ masses, and tested on 500 GeV. 54

9.4 Significance (simplified) for different variations of XGBoost models trained on all LQ masses, and tested on 500 GeV. 57

11.1 Recorded numbers of raw events with and without preselection, where events with negative weights were removed. 67

11.2 Comparison of accuracy, F1 score and AUC of models trained on preselected events only (before) and models trained on events without preselection (after). The displayed values correspond to Figures 9.7, 11.1 and 11.2. 69

12.1 Significance (simplified) for different variations of XGBoost binary classifiers trained on data without preselection, and tested on data with preselection with all LQ masses as signal, LQ $\sigma = 0.001\,\text{pb}$. 74

13.1 Asymptotic expected upper limit at 95% CL on cross-section in pb, with 68% and 95% confidence intervals, obtained with the TabNet model. 76

13.2 Asymptotic expected upper limit at 95% CL on cross-section in pb, with 68% and 95% confidence intervals, obtained with the TabNet model. 76

13.3 Asymptotic expected upper limit at 95% CL on cross-section in pb, with 68% and 95% confidence intervals, obtained with the XGBoost model. 77

13.4 Asymptotic expected upper limit at 95% CL on cross-section in pb, with 68% and 95% confidence intervals, obtained with the XGBoost model. 77

D.1 Feature importance of top 60 features of the selected TabNet model trained on 71 selected features, using all LQ masses, using information from 20 trials. 96

D.2 Feature importance of top 60 features of the selected XGBoost model trained on 71 selected features, using all LQ masses, using information from 20 trials. 97

Introduction

Particle physics is a field of study that seeks to understand the fundamental building blocks of the universe and the forces that govern them. The Standard Model of particle physics is the prevailing theory that describes the behavior of these particles and their interactions. However, there are still many phenomena that cannot be explained by the Standard Model alone, which has led to the hypothesis of new particles that may exist beyond the scope of the Standard Model.

One such particle is the Leptoquark [12], which is a hypothetical particle that is believed to interact with both leptons and quarks, the two types of particles that make up matter. The Leptoquark is postulated to have a range of masses and couplings, which could explain various unexplained phenomena in the Standard Model [12].

The discovery of the Higgs boson in 2012 was a major breakthrough in particle physics, providing insight into the nature of the universe and prompting further exploration into the realm of undiscovered particles. The discovery of the Higgs boson was a long-awaited confirmation of the existence of a particle that had been postulated for decades, and its discovery sent shockwaves throughout the science community due to its implications on the world of particle physics [13].

The discovery of the Higgs boson has also led to increased interest in the search for other undiscovered particles, including the Leptoquark. In this thesis, we focus on the separation of Leptoquark pair-production from background events in the 2lSS+1τ channel. This channel involves events decaying into two light leptons (electron and muon) of the same sign and one hadronically decaying Tau-lepton. The focus on separation of signal from the background is important as it allows us to estimate the upper limit of the cross-section for the pair-production of Leptoquarks.

To achieve this separation, we implement various machine learning models, including neural networks and decision trees. The performance of each model is tested and compared, and the models with the best performance are selected. We analyze the most important features of these models to train smaller and more efficient models. The TRExFitter framework is used to perform statistical tests, which provide additional insights to improve the models' performance.

Part I

Theory & Outline

Chapter 1

State of the Art Research

CERN, the European Organization for Nuclear Research, is a world-renowned research organization located in Geneva, Switzerland. It is dedicated to advancing scientific knowledge in the field of particle physics and is home to the largest and most powerful particle accelerator in the world - the Large Hadron Collider (LHC) [14].

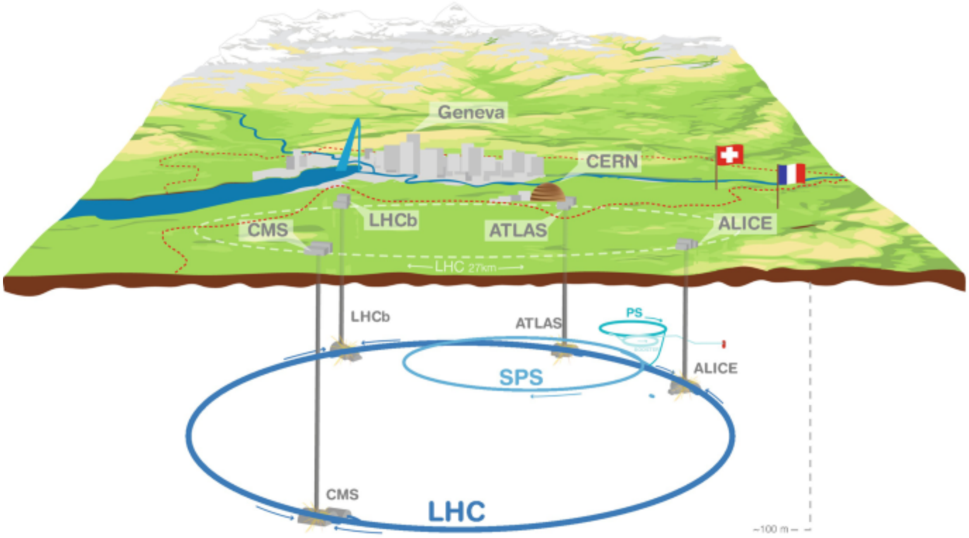


Figure 1.1: Diagram of the Large Hadron Collider with the four largest detectors [1].

The LHC is a ring of superconducting magnets with a circumference of 27 kilometres, where two beams of protons are accelerated close to the speed of light, then deviated to collide in one of the few detectors located around the ring. One of these detectors is the ATLAS detector, which is located 100 meters below ground, and is 46 meters in length, and 25 meters in diameter. It is equipped with six different detection subsystems located in layers around the collision point [15].

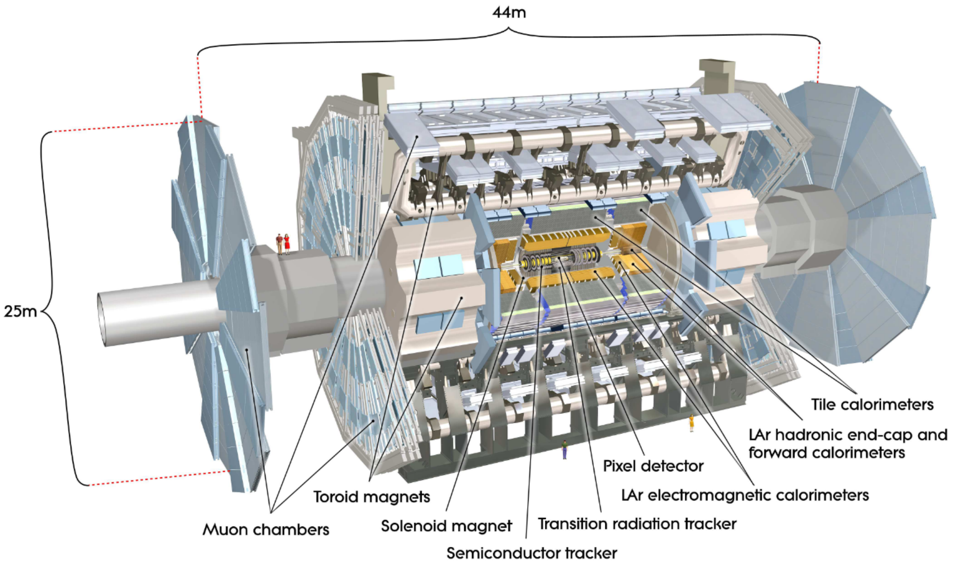

44m
25m
Tile calorimeters
LAr hadronic end-cap and forward calorimeters
Pixel detector
LAr electromagnetic calorimeters
Toroid magnets
Muon chambers
Solenoid magnet
Transition radiation tracker
Semiconductor tracker

Figure 1.2: Cross-section diagram of the ATLAS detector [2].

Massive amounts of data are collected from the millions of interactions that occur in the ATLAS detector every second. CERN collaborates with scientists and researchers from around the world to carry out cutting-edge research in particle physics, computing, engineering, and technology [16].

Chapter 2

Theoretical Background

2.1 Cross-section

In particle physics, the probability of two particles colliding is measured by the cross-section. It is typically denoted as sigma and has units of area which is typically measured in barns (b),

$$1 \text{ barn} = 10^{-24}\text{cm}^2.$$

2.2 Luminosity

Luminosity is the proportionality factor between the number of events per second $\frac{dR}{dt}$ and the cross-section σ and it measures the ability of a particle accelerator to produce a required number of events [17].

$$\frac{dR}{dt} = \mathcal{L}(t) \cdot \sigma \tag{2.1}$$

After integrating Equation 2.1, we obtain Equation 2.2 which can be used to estimate the number of events of a process knowing the luminosity of the collider and the cross-section of the given process.

$$R = \mathcal{L} \cdot \sigma \tag{2.2}$$

Chapter 3

The Standard Model

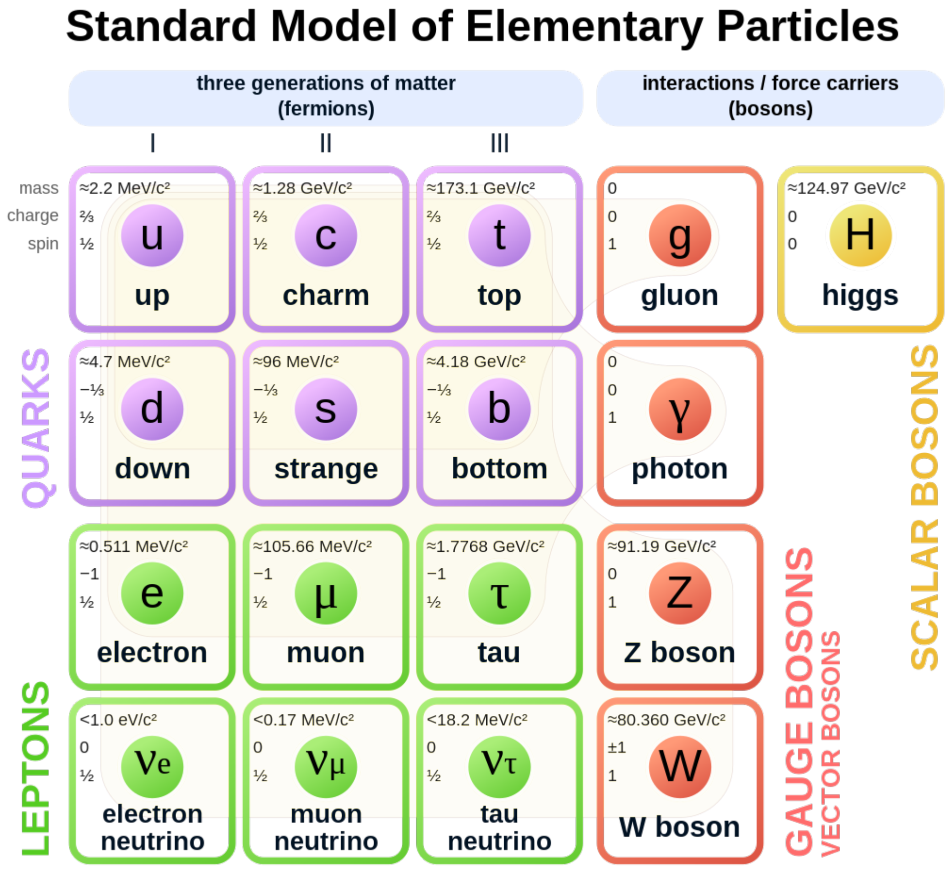


Figure 3.1: Standard Model of Elementary Particles [3].

The universe is composed of different types of particles. The two basic types of particles are quarks and leptons. More information on quarks is given in Section 3.1, and information on leptons is given in Section 3.2.

In addition to matter particles, there is another group of particles responsible for the four fundamental forces in the universe. These force and carrier particles have different ranges, with gluon responsible for the strong force, W and Z bosons responsible for the weak force, photon responsible for the electromagnetic force, and the predicted but yet undiscovered graviton believed to be responsible for gravity [3].

All elementary particles that belong to the Standard Model are shown in Figure 3.1.

3.1 Quarks

Quarks are “elementary particles” that constitute the foundation of matter. They are the building blocks of mesons, which are composed of quark and anti-quark pairs, and baryons, which consist of three quarks. Protons and neutrons, which make up the atomic nucleus, are examples of baryons.

As elementary particles, quarks lack internal structure and cannot be further broken down into smaller particles. Their distinctiveness arises from the six known "flavors" in which they exist: up, down, charm, strange, top, and bottom. Each flavor carries a unique set of quantum numbers, including charge, spin, and flavor, that govern their interactions with other particles [18].

3.2 Leptons

Leptons are also “elementary particles” which are as fundamental as quarks. The Standard Model consists of six leptons – the electron, muon, and tau particles, and their associated neutrinos [19].

Chapter 4

Beyond the Standard Model

4.1 Leptoquarks

A Leptoquark (LQ) is a hypothetical particle consisting of color-triplet bosons with non-zero baryon and lepton numbers. Due to this, LQs can couple to both quarks and leptons. Currently, LQs are placed beyond the Standard Model.

There are three primary modes of LQ production: pair-production, single-production, and off-shell production [4], shown in Figure 4.1.

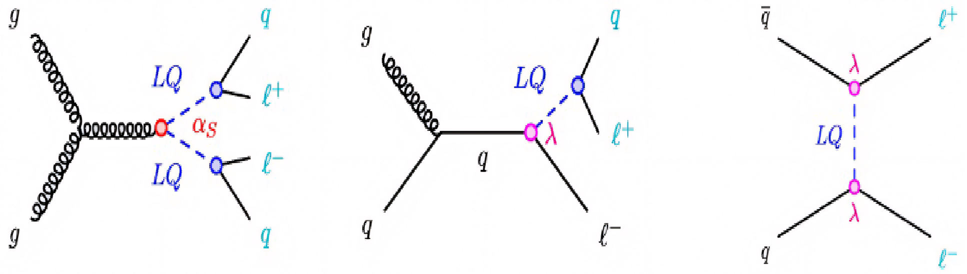


Figure 4.1: Feynman diagrams for Leptoquark pair-production (left), single-production (centre), off-shell production (right) [4].

The final states of the pair-production Leptoquark and Higgs boson $t\bar{t}H$ ($H \to \tau\tau$) decay exhibit a similar pattern, as illustrated in Figure 4.2. This similarity is significant in the context of this analysis, as it allows for the application of similar techniques used in the Higgs boson analysis.

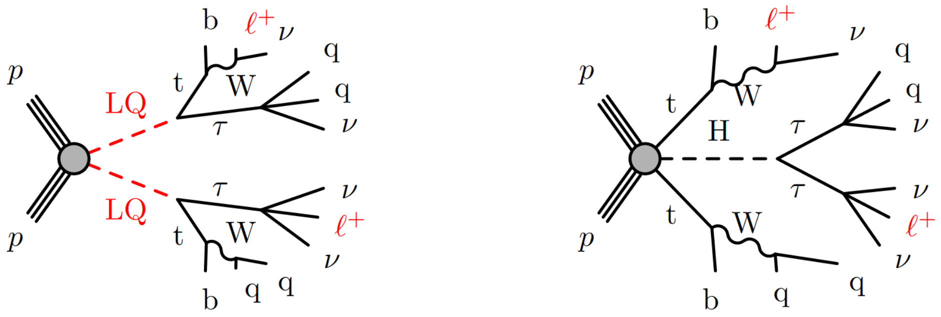


Figure 4.2: Feynman diagrams for the same final state for pair-production mode of Leptoquark (left) and production Higgs boson ($t\bar{t}H(H \rightarrow \tau\tau)$) (right) [4].

In the pair-production process, each LQ decay consists of a top quark and a τ-lepton, one of which is a hadronically decaying τ-lepton and the other one is a leptonically decaying τ-lepton. Moreover, two light leptons (e, μ) are present in the final state. These conditions are included in the preselection.

Chapter 5

Experimental and Simulated Data

5.1 Monte Carlo Simulation

Monte Carlo Simulation is a mathematical technique that is used to estimate the potential results of an event characterized by uncertainty. This method involves creating a model of possible results based on the probability distributions of contributing variables. For a set of inputs, the simulation can calculate or approximate the outcome of an event [20].

Monte Carlo simulations offer several advantages for research at the LHC. First, before an experiment is conducted at ATLAS, it can be simulated based on the Standard Model. This allows for an estimation of the potential outcome of the experiment, including yields of different processes. Comparing the simulated and experimental data can help identify any discrepancies in the Standard Model.

Secondly, Monte Carlo simulations can be used to simulate processes that cannot be described by the Standard Model, such as Leptoquark pair-production. These events can be simulated by physics beyond the Standard Model and then be compared against the rest of the simulated and collected data.

Thirdly, despite the complexity of the software used to run Monte Carlo simulations, they are highly efficient and cost-effective to describe the recorded data.

Finally, it is worth noting that the high number of events generated by Monte Carlo simulations can be especially beneficial for machine learning models which tend to perform better the more data they can learn from [21].

5.2 Processes

Events of different processes on the 2lSS+1τ channel are considered in this thesis, the signal process being the pair-production process of the Leptoquark. As for background, the following processes are considered:

- $t\bar{t}H$ - top quark, anti-top quark, Higgs boson
- $t\bar{t}W$ - top quark, anti-top quark, W boson
- $t\bar{t}Z$ - top quark, anti-top quark, Z boson
- $t\bar{t}$ - top quark, anti-top quark
- VV - diboson ($V = W, Z$)
- "Other" - minor background processes (complete list in Table 5.5)

5.3 Event Selection

To analyze the data collected or simulated during the ATLAS experiment, it is necessary to filter out events that are relevant to the analysis. This process is known as 'preselection', where events are selected based on certain criteria.

The ROOT framework is used to handle both the data storage and data selection processes. More information about the ROOT framework is given in Section 6.1.

For the 2lSS+1τ channel, Table 5.1 displays the preselection criteria used to filter out events that are not relevant to the analysis.

Preselection criteria
2 light leptons of same sign
Exactly one hadronically decaying τ
At least one b-quark jet
4 or more jets

Table 5.1: Summary of preselection criteria. A complete definition of the preselection region is given in Appendix A.

To understand the first criterion, it is necessary to explain how leptons are represented in the data. The types of the first and second leptons are indicated by the values in "lep_ID_0" and "lep_ID_1", respectively. Their possible values and meanings are listed in Table 5.2.

Value	Lepton Type
11	electron
13	muon
-11	anti-electron
-13	anti-muon

Table 5.2: Four different types of lepton represented by four possible values of "lep_ID_0" and "lep_ID_1".

In the preselection region, the condition $(lep\_ID\_0) == 13$ checks whether the first lepton is a muon, the condition $(lep\_ID\_0) == 11$ checks whether the first lepton is an electron, and the condition $((lep\_ID\_0 \cdot lep\_ID\_1) > 0)$ ensures that the leptons have the same sign.

Similarly, for the second condition, the number of τ leptons is stored in a separate variable called "nTaus_OR" and the condition $nTaus\_OR == 1$ ensures that exactly one such τ is present.

5.4 Event Weights

When two protons collide in an experiment, a large number of possible processes can occur, and the probability of each process can vary significantly. While simulated data can replicate these interactions, the proportions of the simulated processes may not be the same as those observed in the actual experiment. To better reflect reality, event weights are used to adjust the simulated data.

Event weights are factors calculated individually for each event as shown in Equation 5.1.

$$w_e = \frac{L \cdot \sigma \cdot \prod w_i}{w_T} \tag{5.1}$$

- w_e - event weight
- σ (xs) - cross-section
- L - luminosity corresponding to the year of production

$$L = \begin{cases} 36646.74, & \text{if } RunYear = 2015 \vee RunYear = 2016 \\ 44630.6, & \text{if } RunYear = 2017 \\ 58791.6, & \text{if } RunYear = 2018 \end{cases}$$

- w_i - Set of features factored in.

Feature Names
custTrigSF_LooseID_FCLooseIso_DLT
weight_pileup
jvtSF_customOR
bTagSF_weight_DL1r_85
weight_mc
lep_SF_CombinedTight_0
lep_SF_CombinedTight_1
lepSF_PLIV_Prompt_0
lepSF_PLIV_Prompt_1

- w_T (*totalEventsWeighted*)- normalization factor (number of simulated events in the ntuple)

5.5 Process Yields

There are two ways to measure the number of events: raw or weighted. Raw measurement simply adds up the occurrences of each process as they are. On the other hand, weighted measurement, also known as yield, applies weights to obtain a more accurate estimate of the number of events, accounting for the probability of their occurrences as explained in Section 5.4.

Depending on the applied preselection cut, the yields may vary greatly. Tables 5.3 and 5.4 list the yields of processes used in this analysis. This analysis is based on version 8 (V8) of the dataset, but for comparison, the yields of the version 6 (V6) dataset are also included.

The production cross-section of event generation is used. The smaller yields for V8 are due to the harder preselection and harder object definition. The theoretical cross-section of the LQ signal used to compute the yields in Table 5.3 were estimated by MadGraph5_aMCNLOPythia8 as documented in [22].

LQ mass [GeV]	Raw events (V8)	Yields (V8)	Raw events (V6)	Yields (V6)
500	2257	232.069	2822	316.221
600	3347	122.717	4087	156.040
700	2929	51.370	2642	45.946
800	2885	23.275	3271	26.230
900	2792	10.135	3216	11.952
1000	1153	1.986	2345	3.981
1100	2638	2.139	2975	2.458
1200	2530	1.026	2869	1.187
1300	1258	0.573	1394	0.621
1400	1213	0.267	1367	0.310
1500	1164	0.143	1264	0.155
1600	1232	0.074	1375	0.083

Table 5.3: Number of raw events and yields for each LQ mass (500 - 1600 GeV), using theoretical cross-sections.

Masses of 500 and 1000 GeV in the V8 dataset have lower yields than they should, due to missing samples. Each mass comprises of samples for different run years which are stored in separate ROOT files. The samples corresponding to 2015 and 2016 ("mc16a") are not present for the mass of 500 GeV, and the samples corresponding to 2015 and 2016 ("mc16a") and 2017 ("mc16d") are not present for the 1000 GeV. To compensate for this fact, the event weight of samples belonging to these two masses must be calculated using the luminosity of the available samples only.

Process	Raw events (V8)	Yields (V8)	Raw events (V6)	Yields (V6)
LQ (all masses)	25398	445.8	29627	565.2
$t\bar{t}H$	15567	12.6	29541	22.8
$t\bar{t}W$	3038	11.2	10248	24.9
$t\bar{t}Z$	8819	12.5	27397	18.2
$t\bar{t}$	20	2.3	181	21.0
VV	1907	4.4	2805	6.5
"Other"	1877	5.0	3142	11.4

Table 5.4: Number of raw events and yield for each process and combination of processes, using theoretical cross-sections.

It is apparent that V8 dataset contains significantly fewer events than V6 of the dataset, particularly for background processes. Moreover, to increase the number of diboson samples, the PLIV cut will be removed for data used in the training of the models.

Each process is stored in a separate ROOT file, named according to the process' dataset ID (DSID), and events of each process are organized by year into different folders. The corresponding DSIDs for each process are listed in Table 5.5. Further information on ROOT files is provided in Section 6.1.

For TRExFitter plots the theoretical cross-section is set to 1 pb. Yields with cross-section set to 1 pb are shown in Table 5.6.

Process	DSID
LQ (all masses)	310175, 313396, 313397, 312244, 312245, 310175, 312246, 312247, 312248, 312249, 312250, 313398
$t\bar{t}H$	346343, 346344, 346345
$t\bar{t}W$	700168, 700205
$t\bar{t}Z$	700309
$t\bar{t}$	410470
VV	363356, 363358, 363359, 363360, 363489, 364250, 364253, 364254, 364255, 364283, 364284, 364285, 364286, 364287
"Other"	304014, 342284, 342285, 364242, 364243, 364244, 364245, 364246, 364247, 364248, 410080, 410081, 410397, 410398, 410399, 410408, 410560

Table 5.5: DSID for each process. LQ DSIDs arranged in order from 500 to 1600 GeV); $t\bar{t}W$ DSIDs consist of ttW2210 and ttW2210_EW samples, respectively; "Other" DSIDs consist of threeTop, VH, VVV, fourTop, rareTop, WtZ and tZ samples in this order.

LQ mass [GeV]	Yield (V8)	Yield (V6)
500	854.8	1164.8
600	1334.2	1696.5
700	1462.7	1308.2
800	1599.9	1803.0
900	1562.3	1842.1
1000	655.9	1314.9
1100	1451.3	1667.5
1200	1369.5	1584.4
1300	1483.5	1607.2
1400	1296.3	1504.1
1500	1284.8	1394.3
1600	1215.6	1359.2

Table 5.6: Yield for V6 and V8 dataset for each LQ mass (500 - 1600 GeV), with LQ $\sigma = 1\,\text{pb}$).

Chapter 6

Frameworks

6.1 ROOT Framework

As shown in previous chapters, high-energy physics involves working with large amounts of data in real-time, which necessitates the use of specialized technologies. For this reason, the ROOT framework was developed at CERN and is currently utilized by researchers worldwide. ROOT, which is written in C++, offers a wide range of features from data storage and plotting graphs and histograms to Monte Carlo event generation and distributed computing [23].

ROOT offers multiple data structures, all of which are saved as binary objects in a ROOT file format. The most powerful available data structure is a *tree*. This nested data-structure may consist of *branches*, *leaves* but also other *trees*. A *leaf* is always the end point of a branch and is an equivalent of a variable.

The benefit of the *tree* structure is that it allows a tailored representation of a dataset from simple representations of tables to complex multi-branch structures.

6.1.1 Uproot

ROOT comes with an interpreter which can be called via `root` from the terminal. There are many available functions in ROOT. For example, `TFile::Open()` can be used to open a ROOT file, `TFile::Get()` to load an object (such as a tree) from the loaded file and `TGraph::Draw()` to produce a simple plot of its values.

However, given that the rest of the architecture is built in Python it is more suitable to use ROOT in the form of Python library called "Uproot". Uproot is specifically needed for the purpose of reading and writing ROOT files, whereas for the remainder of the analysis, CSV, NPY and PKL files are used instead due to their simplicity and ease of use.

6.2 Python

All of the development was done remotely on LXPLUS and CERNBox platforms. This was done because of the unprocessed NTuples taking up approximately 3 TB of space, including the systematic NTuples. The coding was implemented using Python 3.6.8 and a long list of libraries, such as numpy, pandas and scipy. A complete list of used libraries is given in Appendix B.

6.3 Optuna

Optuna is a hyper-parameter optimization framework that aims to find the optimal combination of hyper-parameters for a given machine learning model. To achieve this, Optuna [24] creates a space of hyper-parameter values and uses the Tree-Structured Parzen Estimator (TPE) algorithm to evaluate their effect on the model's performance, based on a user-defined "objective" function. In this analysis, the simplified formula for significance will be used as the performance metric in the objective function.

To run Optuna, a study must be created that includes information about the model to be optimized, the objective function to be used for evaluation, and the set of hyper-parameters. The study then iteratively searches through the hyper-parameter space. During each iteration it evaluates the model's performance using the objective function and updates the hyper-parameters based on the TPE algorithm. This is done for a specified number of trials or until the optimal hyper-parameters are found [24].

6.4 TRExFitter

TRExFitter stands for "Template fits for Reduced data eXperiment" and it is a software framework used in high-energy physics for statistical analysis, likelihood estimation and uncertainty estimation among many others [25]. The two particular use cases of TRExFitter in this analysis are production of histogram plots for preselected and classified data and the estimation of the upper limit of cross-section of the LQ.

A thorough guide to TRExFitter installation is available in the TRExFitter README [26].

In order to run a TRExFitter job, firstly, TRExFitter must be compiled using `source setup.sh` located inside the TRExFitter repository [26]. After this, multiple commands become available, including `trex-fitter`. Secondly, a configuration file must be prepared which tells TRExFitter what needs to be done.

The configuration file is a plain text file that is divided into blocks, where each block has a set of parameters:

- Job - general options

  NtuplePaths - location of the root files

  NtupleFiles - name of the NTuple root files

  MCweight - specifies the event weight formula. It can be applied either to all or individual samples.

  Lumi - value of luminosity

  BlindingThreshold - maximum allowed signal to background ratio per bin to ensure that the results remain blinded

  ReplacementFile - location and name of the replacement file which contains placeholders. Before a *config* file is evaluated, its placeholders that begin with "XXX" are replaced with values of corresponding placeholders from the replacement file.

- Region - distributions used in the fit and specifies variables and cuts

  Selection - criteria used for the cut

  LogScale - whether logarithmic scale is used

- Sample - Defines samples of all included processes (signal and background)

  Group - samples can be grouped in order to combine samples of multiple processes into one

- Systematic - specifies systematic uncertainties

Lastly, a TRExFitter job can be run as follows:

```
trex-fitter <actions> <config file> <options>
```

For more detailed information see TRExFitter Template fits [27].

Chapter 7

Machine Learning

7.1 Neural Network

With the rise of artificial intelligence, many different machine learning algorithms have been developed and researched. Neural networks are a subset of machine learning and can be used for both supervised and unsupervised learning.

In this analysis, supervised machine learning is utilized. Therefore, to train supervised models, the expected outut (or labels) must be provided during the training process.

7.1.1 Artificial Neuron

The smallest component of a Neural Network (NN) is a neuron. A neuron is a function f_j applied on an input vector $x = (x_1, ..., x_d)$ multiplied by a vector of weights $w_j = (x_{j,1}, ..., x_{j,d})$, and added to a neuron bias b_j[28].

$$y_j = f_j(x) = \phi(\langle w_j, x\rangle + b_j) \tag{7.1}$$

In Equation 7.1, y_j is the output of the function and ϕ is a non-linear activation function which is essential for the separation of non-linear data.

A diagram of this process is shown in Figure 7.1.

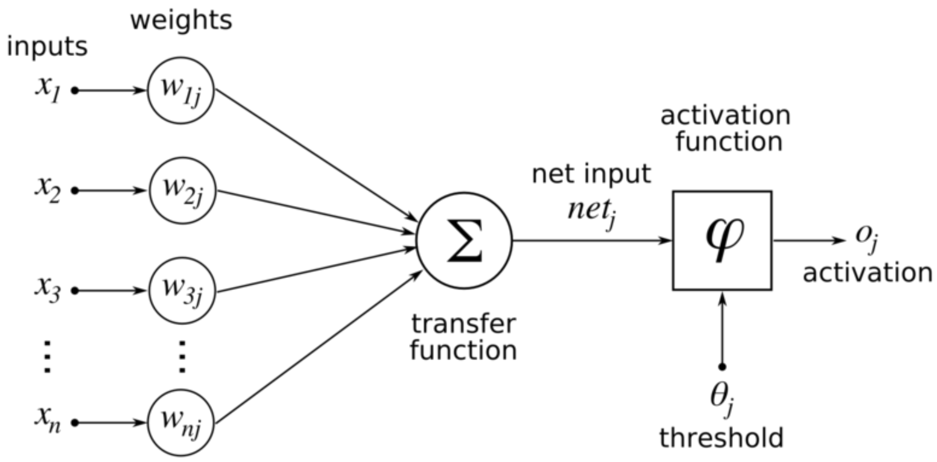


Figure 7.1: Role of a single neuron [5].

Examples of activation functions include:

- The sigmoid function

$$\phi(x) = \frac{1}{1+e^{-x}} \tag{7.2}$$

- The hyperbolic tangent function ("tanh")

$$\phi(x) = \frac{e^{2x}-1}{e^{2x}+1} \tag{7.3}$$

- The Rectified Linear Unit (ReLU)

$$\phi(x) = max(0, x) \tag{7.4}$$

ReLU was chosen because of its superior efficiency, particularly in the back-propagation step in which the derivative of the activation function needs to be computed. While the derivative of the sigmoid function 7.2 has infinite variations on the interval $x \in \langle -\infty, \infty \rangle$, ReLU 7.4 only has three:

$$\frac{\partial \phi}{\partial x} = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{if } x > 0 \\ "undefined", & \text{if } x = 0 \end{cases}$$

7.1.2 Multilayer Perceptron

A multilayer perceptron (neural network) is an architecture that consists of several hidden layers of neurons where the output of a neuron in one such layer becomes the input of another neuron in the next layer, as shown in Figure 7.2. Other types of connections may exist where the output of one neuron can be fed as the input of a neuron in the same layer which is the case for recurrent neural networks. Depending on the purpose of the network - regression or classification - a different activation function is used in the output layer.

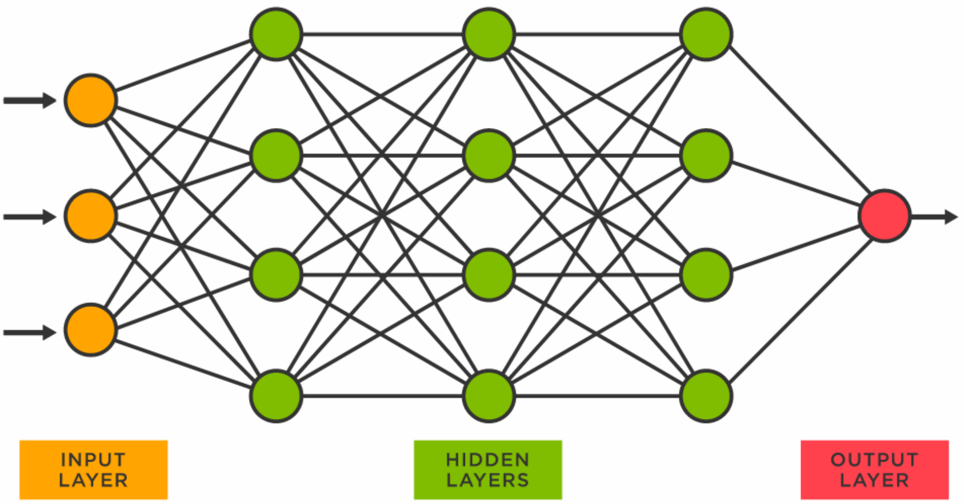


Figure 7.2: Structure of a neural network [6].

Separation of signal from background can be performed either via a binary classifier or a multi-class classifier. In this analysis the latter is used as it provides more information about the classified data.

Multi-class classification contains one output neuron per class k. The resulting output is a probability prediction $\mathbb{P}(Y = k/X)$ for each of the classes, such that $\sum\limits_{k} \mathbb{P}(Y = k/X) = 1$ [28].

For this reason the softmax function is used,

$$softmax(z)_k = \frac{e^{z_k}}{\sum\limits_{j} e^{z_j}}. \tag{7.5}$$

7.1.3 Loss Function

The expected loss L of the network measures the classification error.

Similarly, for multi-class classification, a suitable loss function must be chosen. For this purpose, negative log-likelihood loss (provided by Pytorch) is selected. In its irreducible form, it is defined as:

$$\ell(x, y) = \{l_1, ..., l_N\}^T, l_n = -w_{y_n} \cdot x_{n,y_n}, \tag{7.6}$$

where x is the input, y is the target output, w is the weight, and N is the batch size [29].

7.2 Bayesian Neural Network

Multi-layer perceptron uses a deterministic weight system, and these weights are incrementally adjusted over the course of training. One downside of this deterministic approach is that such models may be prone to over-fitting. As a result, other methods - such as the use of dropout layers - need to be employed to reduce the likelihood of overfitting.

Dropout layers introduce a probability that neurons which connect to that layer will be "dropped out" during the forward-pass part of the algorithm. This way the network reduces the amount of information from the training data which may improve overfitting.

However, information removed by the dropout may prove useful to the model, and therefore other methods of preventing over-fitting have been developed, including stochastic sampling of weights. Using a Bayes linear layer as opposed to a traditional linear layer, event weights are sampled using a distribution that is trained over time,

$$q_{\mu,\sigma}(w_i) = \mathcal{N}(\mu_i, \sigma_i^2), \tag{7.7}$$

where the distribution of weight w_i is given as a normal distribution with parameters mean μ and variance σ^2 which are initially equal for all weights. Over time, as the model learns, these parameters are adjusted to better reflect the optimal distribution of each weight. More information on the implementation of Bayesian Neural Networks can be found in the bayesian-neural-network-pytorch repository [30].

7.3 Decision Trees

A decision tree is a simple, yet powerful classifier expressed as a hierarchical model of decisions and their consequences. The decision tree consists of a root node called "root", internal nodes which have both incoming and outgoing edges, and "leaf" nodes which act as terminal nodes, as shown in Figure 7.3. Each internal node divides the decision space into two or more subspaces. This way data can be classified on the basis of discrete criteria [31].

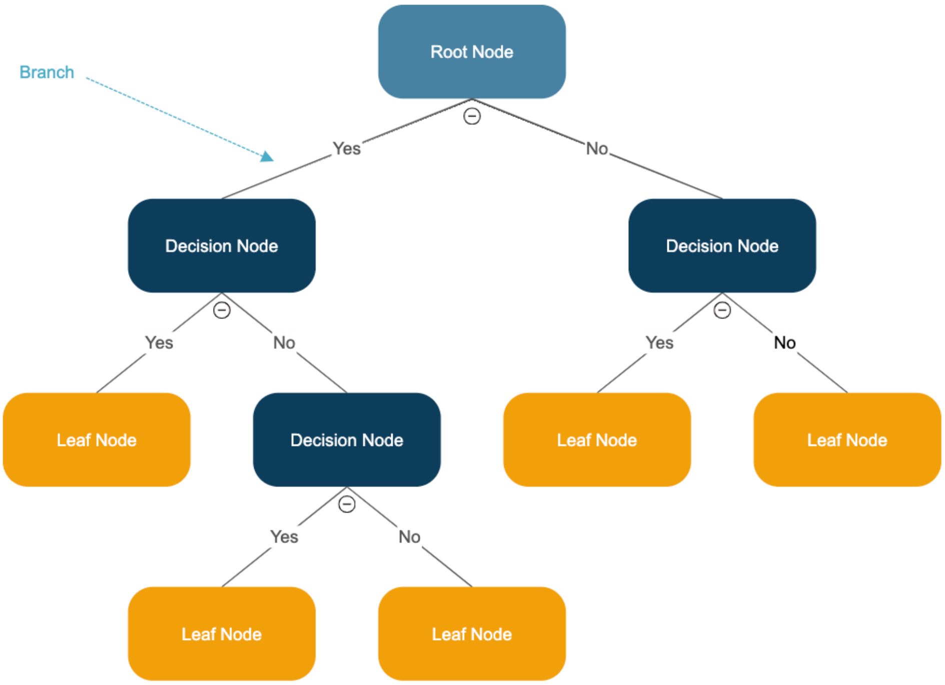


Figure 7.3: Structure of a decision tree [7].

7.3.1 Ensemble Methods

Bias Variance Decomposition

There are two extremes of the complexities of the model. If a model is too simple, it will likely underfit, leading to poor performance in both training and test data. This behavior is reflected as a high value of the "bias" term. On the other hand, a complex model may overfit, and thus achieve high performance on the training data but low performance on test data. This would result in a high "variance" [32]. An ideal model would have low bias and low variance.

For the mean squared error (MSE), the decomposition of bias and variance is given below. For samples x predictions y are produced. y_* is the optimal deterministic prediction and t is a random sample of the true conditional $p(t|x)$,

$$\mathbb{E}[(y-t)^2] = (y_* - \mathbb{E}[y])^2 + Var(y) + Var(t). \tag{7.8}$$

Furthermore, definitions of each component in Equation 7.8 are:

- Bias = $(y_* - \mathbb{E}[y])$
- Variance = $Var(y)$
- Noise = $Var(t)$

Equation 7.8 gives one more term $Var(t)$ called "Bayes error" or noise which arises from the randomness of the process which produced the data. Noise is irreducible because of its independence from the performance of the model.

7.3.2 Bagging

One way to reduce variance is by employing bagging. Bagging uses a bootstrapping re-sampling technique that creates subsets of the original dataset by randomly selecting samples with replacement. A set of weak classifiers is then independently trained on each subset. Finally, when a sample needs to be classified, the results of classification from all weak classifiers are averaged to produce the final prediction. The combination of weak classifiers is referred to as a strong classifier [33].

Since each weak classifier in bagging uses only a subset of the original dataset, the strong classifier is less likely to over-fit. This results in lower variance because the average of the individual variances tends to be lower than without bagging. Thus, bagging helps to minimize over-fitting and reduce variance.

7.3.3 Boosting

Boosting is another ensemble method where a series of weak classifiers is trained sequentially, focusing on samples that were misclassified by the classifiers before. This is done by initializing each training sample with a uniform sample weight. A weak model is then trained on the training data, and the error for each misclassified sample is calculated. Using a computed error, weights of all the samples are adjusted in such way that the misclassified samples receive more weight. This process is then repeated with the next set of weak classifiers giving the previously misclassified samples more importance [33].

From the approach it is clear that the goal of boosting is to reduce underfitting, as with each iteration the weak classifiers are encouraged to classify the training set as accurately as possible. This typically results in a strong model with very low bias.

7.4 XGBoost

As described in the XGBoost derivation process, for a dataset $D = (x_i, y_i) : i = 1...n, x_i \in \mathbb{R}^m, y_i \in \mathbb{R}$, for n samples and m features, a prediction $\hat{y}_i$ for a sample x_i is defined as,

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), f_k \in F. \tag{7.9}$$

The prediction score of the k-th tree f_k for the i-th sample x_i is denoted as $f_k(x_i)$.

The XGBoost learning process is guided by the objective function 7.10 that it tries to minimize. It consists of the loss function l which measures the differences between the prediction $\hat{y}_i$ and the target value y_i. Therefore, the better the model, the smaller the difference between the two values.

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{7.10}$$

The term Ω penalizes the complexity of the model in order to prevent overfitting.

$$\Omega(f_k) = \gamma \cdot T + \frac{1}{2}\lambda ||w||^2 \tag{7.11}$$

As shown in Equation 7.11, its computation involves degrees of regularization γ and λ, the number of leaves T and the scores of leaves w.

For the iterative learning process of the ensemble model, Equation 7.10 can be rewritten and simplified using the second-order Taylor expansion, producing Equation 7.12,

$$Obj^{(t)} \approx \sum_{i=1}^{n} \left[g_i f_i(x_i) + \frac{1}{2} h_i f_i(x_i)^2 \right] + \gamma \cdot T + \sum_{j=1}^{T} \frac{1}{2} \lambda w_j^2, \tag{7.12}$$

where $g_i = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ and $h_i = \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ are the first and second-order partial derivatives of l.

The optimal weight w_j^* of the leaf j and the corresponding optimal value can be calculated:

$$w_j^* = -\frac{G_j}{H_j + \lambda} \tag{7.13}$$

$$Obj^* = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \lambda \cdot T, \tag{7.14}$$

where $G_j = \sum_{i \in \{i|q(x_i)=j\}} g_i$ and $H_j = \sum_{i \in \{i|q(x_i)=j\}} h_i$, denoting the sum of gradients of the indices of the j-th leaf.

In order to find the optimal split point which will maximize the reduction in the loss function, the gain formula is used,

$$G = -\frac{1}{2}\left[\frac{(\sum_{i \in I_L} g_i)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{(\sum_{i \in I_R} g_i)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{(\sum_{i \in I} g_i)^2}{\sum_{i \in I} h_i + \lambda}\right] - \gamma. \tag{7.15}$$

The gain in the objective function consists of four terms corresponding to the left leaf (I_L), right leaf (I_R) and the original leaf I and the regularization term γ. More details are given in [34].

7.4.1 XGBoost Hyperparameters

- learning_rate - controls the amount by which weights are updated during each iteration
- max_depth - maximum allowed depth of the decision tree
- colsample_bytree - fraction of features considered at each split
- min_child_weight - minimum weight required to create a new node during training
- n_estimators = maximum number of trees
- subsample = fraction of training data that is sampled for one iteration of training

A complete list of hyperparameters is given in the XGBoost documentation [35].

7.5 TabNet

TabNet is a machine learning algorithm introduced in 2019 [8] for the construction of deep neural networks (DNN) for classification and regression problems involving tabular data. TabNet's focus on tabular learning could make it very beneficial in this analysis as large datasets are used.

The architecture of a TabNet model resembles a neural network, with some key differences; the main difference being its focus on feature selection. Instead of all features being fed to the model, a TabNet model selects different features in each of its steps (or layers) based on their importance for prediction. This way the model focuses on most important features instead of spending its learning capacity on unnecessary ones.

Firstly, the complete set of features is passed to a feature transformer consisting of:

- Fully-connected layer - applies non-linear transformation to the input data
- Batch normalization - normalizes the transformed data
- Gated linear unit - activation function which produces the strength of inputted features between [0,1] for each feature, where values closer to 1 signify high relevance

The attention transformer selects a small subset of features and produces a mask $M[i]$ using the output of the previous step $[i-1]$. The mask is computed using the sparse-max function as shown in Equation 7.16.

$$M[i] = sparsemax(P[i-1]) \cdot h_i(a[i-1]), \tag{7.16}$$

where h_i is a trainable function of the feature transformer and $P[i]$ is the prior scale term, which signifies how much a particularly feature has been used previously:

$$P[i] = \sum_{j=1}^{i}(\gamma - M[j]), \tag{7.17}$$

where γ is a relaxation parameter. For $\gamma = 1$, each feature is only allowed in one decision step, and the higher the γ the more allowance is given to use a feature in multiple decision steps.

The next feature transformer then multiplies the computed mask by the features f, thus extracting important features. For this subset of features it then produces two outputs: decision $d[i]$ and information for the subsequent step $a[i]$, as shown in Equation 7.18.

$$[d[i], a[i]] = f_i(M[i] \cdot f) \tag{7.18}$$

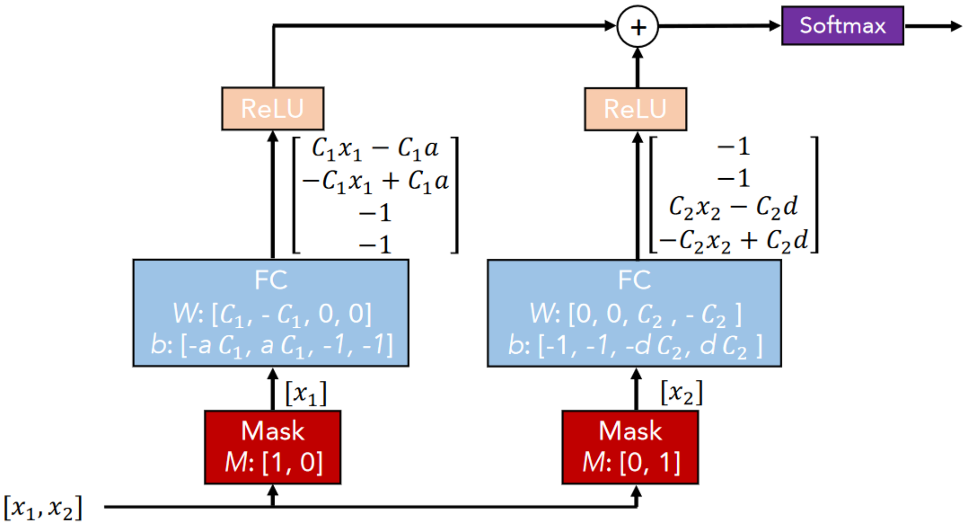

+
Softmax
ReLU
ReLU
$\begin{bmatrix} C_1x_1 - C_1a \\ -C_1x_1 + C_1a \\ -1 \\ -1 \end{bmatrix}$
$\begin{bmatrix} -1 \\ -1 \\ C_2x_2 - C_2d \\ -C_2x_2 + C_2d \end{bmatrix}$
FC
W: [C_1, - C_1, 0, 0]
b: [-a C_1, a C_1, -1, -1]
FC
W: [0, 0, C_2 , - C_2]
b: [-1, -1, -d C_2, d C_2]
[x_1]
[x_2]
Mask
M: [1, 0]
Mask
M: [0, 1]
[x_1, x_2]

Figure 7.4: Computation of the output [8].

The output d_{out} of the model is an aggregate of all individual decision steps d_i passed through the ReLU function (7.4), the computation of which is shown in Equation 7.19.

$$d_{out} = \sum_{i=1}^{N_{steps}} ReLU(d[i]) \tag{7.19}$$

Details are given in [8].

7.5.1 TabNet Hyperparameters

The learning process and certain properties of the model can be adjusted by hyperparameters. Understanding the purpose of each hyperparameter may help with the tuning process of the model.

Hyperparameters that are given to Optuna for optimization are listed below:

- gamma (γ) - controls the sparsity of the attention mask

  $\gamma = 1$ - each feature can only be used in one step

  $\gamma > 1$ - the higher the gamma, the more likely it is for features to be used in multiple steps

- lambda_sparse - determines how much focus will be placed on important features

  high value - puts more focus on important features

  low value - allows more less-important features to be included

- mask_type - determines which function is used for the attention masks

  "sparsemax" - gives higher weight to a smaller number of features to contribute to the output

  "entmax" - distributes contribution of features to the output more conservatively

- momentum - controls the learning rate of the model

  high value - learning is done gradually, allowing for slower but smoother weight updates

  low value - learning is done rapidly with larger weight updates

- n_a - number of attention steps

  high value - may improve accuracy but is computationally heavy

- n_shared - number of decision steps

  high value - may improve accuracy but is computationally heavy

- n_steps - number of steps

  high value - may improve accuracy but is computationally heavy

- patienceScheduler - sets the maximum number of allowed epochs in which an improvement in validation loss must be observed, otherwise learning is stopped

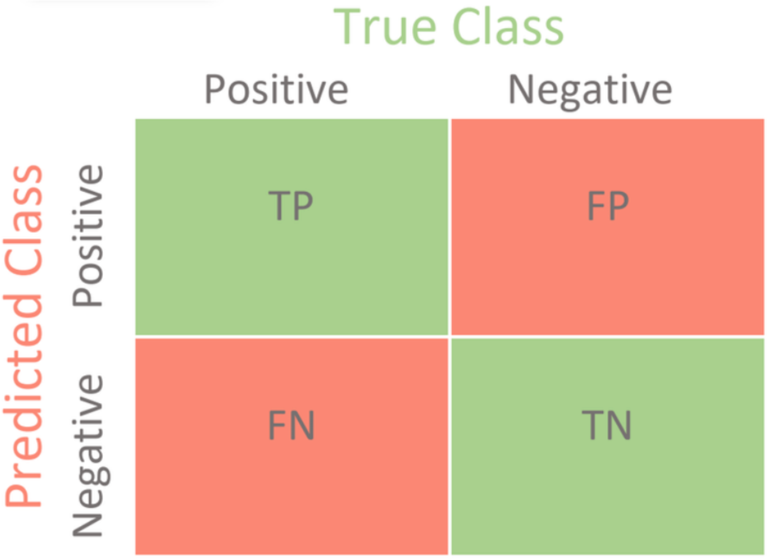
A complete list of hyperparameters and their meaning is give in the TabNet documentation[36].

7.6 Performance Metrics

Given the number of existing machine learning approaches, it is important to have methods to measure and compare their performance. In this section, such metrics will be explained.

7.6.1 Confusion Matrix

After a model has been trained, it needs to be tested on an independent dataset. For a binary classification problem, a test sample can either be classified as 0 or 1, where each digit represents one of the classes which in the context of this analysis are signal (S) and background (B). Since a sample of each class can be misclassified as a sample of the other class, there are four different cases to consider as visualized in Figure 7.5.



		True Class	
		Positive	Negative
Predicted Class	Positive	TP	FP
	Negative	FN	TN

Figure 7.5: Confusion matrix for binary classification [9].

- True Positive (TP) - Correct positive predictions (signal classified as signal)
- True Negative (TN) - Correct negative predictions (background classified as background)
- False Positive (FP) - Incorrect positive predictions (background classified as signal)
- False Negative (FN) - Incorrect negative predictions (signal classified as background)

This is the basis for the following metrics.

Accuracy

Accuracy is one of the most straightforward metrics. It is expressed as a ratio of the correctly classified samples over all test samples. The higher the accuracy the more samples were correctly classified,

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{7.20}$$

ROC Curve

The receiver operating characteristic (ROC) curve is one such performance metrics. The ROC curve displays the properties of the classifier at a range of cut-off points (or thresholds). As a whole, the curve describes the relationship between the true-positive rate (TPR) and the false-positive rate (FPR) which are defined as:

$$TPR = \frac{TP}{TP + FN} \tag{7.21}$$

$$FPR = \frac{FP}{TN + FP}. \tag{7.22}$$

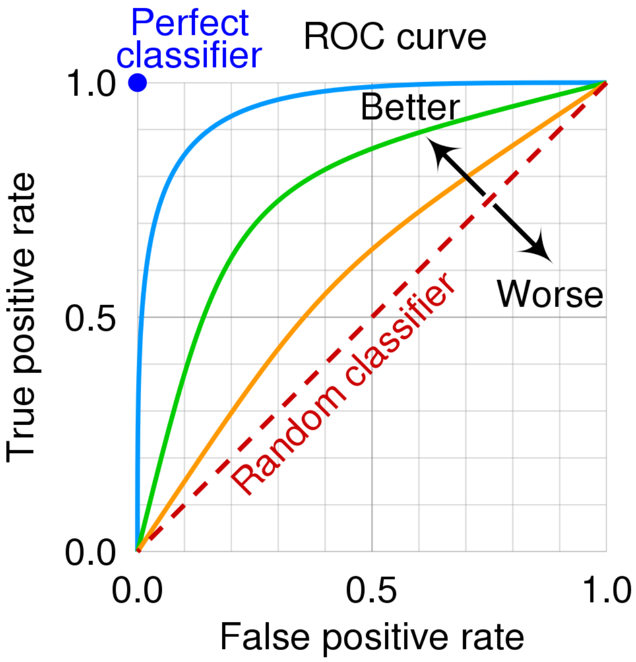


Figure 7.6: ROC curve with additional annotations [10].

The optimal classifier would achieve perfect separation of the signal from the background, resulting in $TPR = 1$ and $FPR = 0$. This state corresponds to the point $[0, 1]$ in Figure 7.6. On the other hand, a random classifier would achieve no separation of signal from background which is illustrated by the red dashed line.

AUC

The overall performance of the classifier across all thresholds can be represented by the area under the ROC curve (AUC). The perfect model would score $AUC = 1$ while a random classifier would obtain $AUC = 0.5$.

For multiclass classification, a combined AUC for all classes can be produced. This is done by treating each class as a signal while the other classes are considered background during the ROC curve computation. This is repeated for all classes. Lastly, a combined AUC can be calculated as a weighted average of AUCs of all classes [37].

F1 Score

$$F1\,Score = \frac{2TP}{2TP + FP + FN} \tag{7.23}$$

The F1 score, defined in Equation 7.23, is beneficial for situations where the reduction of false positives and false negatives is of similar importance [38].

7.7 Significance

In high-energy physics, the separation of the signal process is measured by significance. Significance - symbolised by η - measures the ratio of signal to the square root of background [39], as expressed in Equation 7.24. In this case signal stands for true positives, and background stands for false positives,

$$\eta = \frac{S}{\sqrt{B}}. \tag{7.24}$$

For situations where the background is significantly smaller than the signal, the formula in Equation 7.25 is a better measure.

$$\eta = \frac{S}{\sqrt{S + B}} \tag{7.25}$$

Because of its importance, significance must be considered during model evaluation as one of the most important metrics. Moreover, the threshold where significance is maximized can be selected, giving us the operating point of our model.

Lastly, significance can be used for the expected upper limit of Leptoquark cross-section using TRExFitter. In this approach the significance in each bin of the network output is taken into account.

7.7.1 Confidence Level

In statistics the confidence level represents the degree of certainty that a measurement or observation will meet expectations. This value is measured as a percentage. In the context of our analysis the expected upper limit of Leptoquark cross-section needs to be estimated on a 95% confidence level (CL).

Part II

Implementation

Chapter 8

Data Processing

8.1 Data Selection

To start the process of training the classifiers, the first step is to extract the necessary data out of the ROOT files. However, before any data can be extracted, it must be known which features to extract the data for. Initially a list of 90 features was chosen, containing features that were used in the previous analysis [11]. However, upon closer inspection, some features contained little to no information or had a high number of erroneous values and were therefore removed.

Removed features include features used in preselection, such as "nTaus_OR", "lep_isolationLoose_VarRad_0" or "lep_isolationLoose_VarRad_1", that have constant values for all processes. Other features contained a high number of erroneous values. This is the case for "lep_Mtrktrk_atConvV_CO_0" for which approximately 34% of the data were erroneous (-999). Lastly, a small numbers of features was removed because they had very similar distributions across all processes, for example "Mll012" and "Mll0123". After this cleaning process, 71 features were left. The complete list of features is given in Appendix D.

Process	**Events before**	**Yield before**	**Events after**	**Yield after**
LQ (all masses)	25398	15570.8	20167	20701.6
$t\bar{t}H$	15567	12.6	15377	12.7
$t\bar{t}W$	3038	11.2	2775	13.1
$t\bar{t}Z$	8819	12.5	8432	13.6
$t\bar{t}$	20	2.3	20	2.3
VV	1907	4.4	1897	5.5
"Other"	1877	5.0	1813	5.0

Table 8.1: Recorded number of raw events and yield before and after events with negative weights were removed, with LQ $\sigma = 1\,\text{pb}$.

Using the Uproot library, features from events corresponding to the preselection cut (defined in Section 5.3) were extracted from the ROOT trees and stored in CSV files from which the training and testing datasets are later produced. During this process the event weights (defined in Section 5.4) were calculated and stored separately from the data. It is important to note that some events had to be removed due to having negative weights, which is an artifact of Monte Carlo sample production.

Moreover, Table 8.1 shows that while the number of raw events decreased, the yields of the samples increased. To ensure that correct yields are used during testing, the test set will not have any events removed.

8.2 Training & Testing Datasets

In order to evaluate our trained models, an independent dataset needs to be used. For this reason a 80-20 train-test split is done, meaning that 80% of the original dataset will be used for training and 20% for testing. Furthermore, a validation set is produced from the training set to be used for the tuning of the machine learning models.

The test subset, on the other hand, needs to be adjusted for each of the 12 LQ masses, such that only samples for that particular mass are present for the signal class. This way the acquired models can be tested on each LQ mass separately.

Chapter 9

Selection of Best Models

9.1 Code

All model preparation and manipulation has been done using the code that has been previously worked on by Lukáš Viceník [11]. Using the Git version control system, numerous branches have been created for work with the given models. The current state of the code can be found in `https://gitlab.cern.ch/andre/leptoquarks`. In order to keep the code clean and legible, different parts of the analysis are implemented on different branches in the repository.

In addition, all the TRExFitter configuration files, replacement files and scripts for preparation of the dataset are stored in `https://gitlab.fel.cvut.cz/bohmjani/leptoquark-processing`.

9.2 Choice of Mass for Testing

In order to fairly compare the models, the dataset must be the same. Moreover, due to the differences in signal to background ratio between masses, only one mass is chosen. From the results of the analysis by Lukáš Viceník [11], it is apparent that the level of separation tends to increase the higher the mass of the LQ signal. Therefore, to put the models to the test, the lowest available mass of 500 GeV was selected.

For the production of distributions, all weights of the LQ signal are scaled by a factor of 0.01 pb, in order to be within the same order of magnitude as the background.

Using the values from Table 9.1, we can obtain the baseline value of significance which corresponds to no separation of signal and background,

$$\eta = \frac{1.678}{\sqrt{2.469 + 2.370 + 2.455 + 0.845 + 0.895 + 0.994}} \approx 0.53. \tag{9.1}$$

Process	Combined weight
LQ (all masses)	1.678
$t\bar{t}H$	2.469
$t\bar{t}W$	2.370
$t\bar{t}Z$	2.455
$t\bar{t}$	0.845
VV	0.895
"Other"	0.994

Table 9.1: Combined weight of events in the test set for each background and signal with mass of 500 GeV, with LQ $\sigma = 0.01\,\text{pb}$.

9.3 Neural Network

Multiple MLP (explained in Section 7.1.2) and Bayesian MLP (explained in Section 7.2) models were trained and evaluated, using different combinations of the following parameters.

- number of hidden layers $\in \{1,2,3\}$
- dimension of hidden layers $\in$ {([20, 20]), ([10, 25],[25, 10]), ([20, 50],[50, 20]), ([20, 50], [50, 50], [50, 20]), ([32, 128],[128, 32])}
- hidden layer activation function = ReLU
- dropout (after every hidden layer) $\in \{0, 0.2\}$
- output layer function = LogSoftmax

Type	Layer dimension	Dropout	Significance
MLP	[10, 25],[25, 10]	0	0.955
MLP	[20, 50],[50, 20]	0.2	0.612
MLP	[20, 50],[50, 20]	0	0.894
MLP	[32, 128],[128, 32]	0	0.734
Bayesian MLP	[20, 20]	0	0.875
Bayesian MLP	[10, 25], [25, 10]	0	1.019
Bayesian MLP	[20, 50], [50, 20]	0	1.145
Bayesian MLP	[32, 128], [128, 32]	0	1.108
Bayesian MLP	[20, 50], [50, 50], [50, 20]	0	0.681

Table 9.2: Significance (simplified) for different variations of MLP and Bayesian MLP models trained on all LQ masses, and tested on 500 GeV.

Table 9.2 shows the improved performance when Bayesian linear layers are employed. Secondly, the sweet spot for the number of hidden layers appears to be two for both algorithms, and while MLP favors smaller layer dimensions, performance of the Bayesian MLP models is more levelled across different layer dimensions. Lastly, due to its poor contribution to separation results, only one model with dropout layers was trained. However, this model achieved the poorest performance of all, with little to no separation of signal and background.

9.4 TabNet

One downside of TabNet (explained in Section 7.5) was the relatively long training time compared to the other models. While MLP and XGBoost took under an hour for all combinations of hyperparameters, the TabNet training often took over 90 minutes for simpler architectures and multiple hours for more complex ones. For that reason the Optuna hyper-optimization framework was used for the estimation of the TabNet parameters.

In Optuna, a pruner is used to prune (or terminate) trials which do not achieve a sufficient objective value. This is indirectly controlled by the parameter `N_warmup_steps` which is set to four, meaning that each trial may be pruned if it performed worse than the previous trials after the same number of steps during gradient descent.

A mass of 700 GeV was selected for the Optuna studies as there as more corresponding signal events than for 500 GeV.

The performance of different combinations of parameters is illustrated in Figure 9.1. The darker the line, the better the performance of the trial with the corresponding parameters. The importance of each parameter is displayed in Figure 9.4.

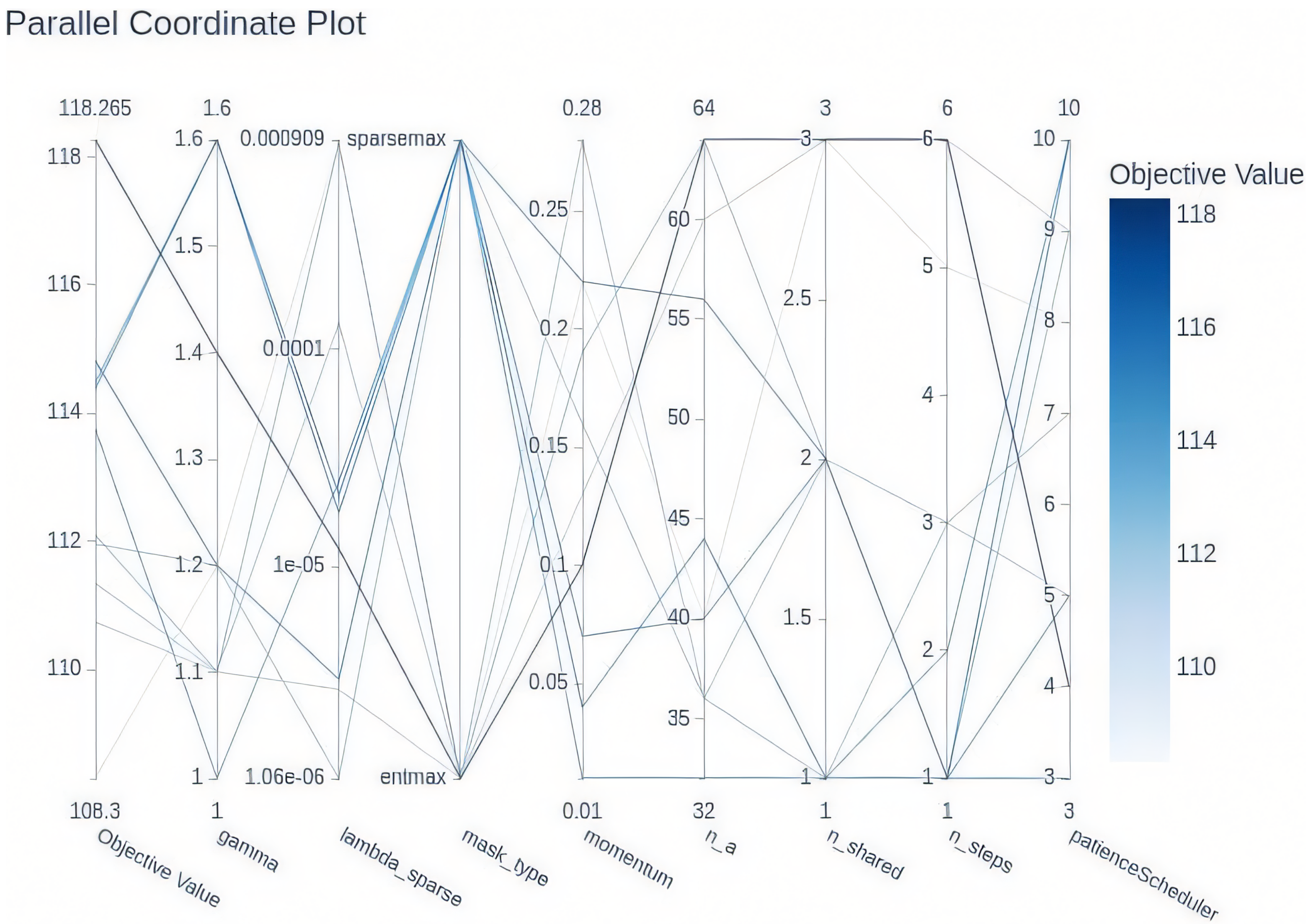


Figure 9.1: Optuna coordinate plot for TabNet trained on all LQ masses and tested on mass of 700 GeV, with LQ $\sigma = 1\,\mathrm{pb}$.

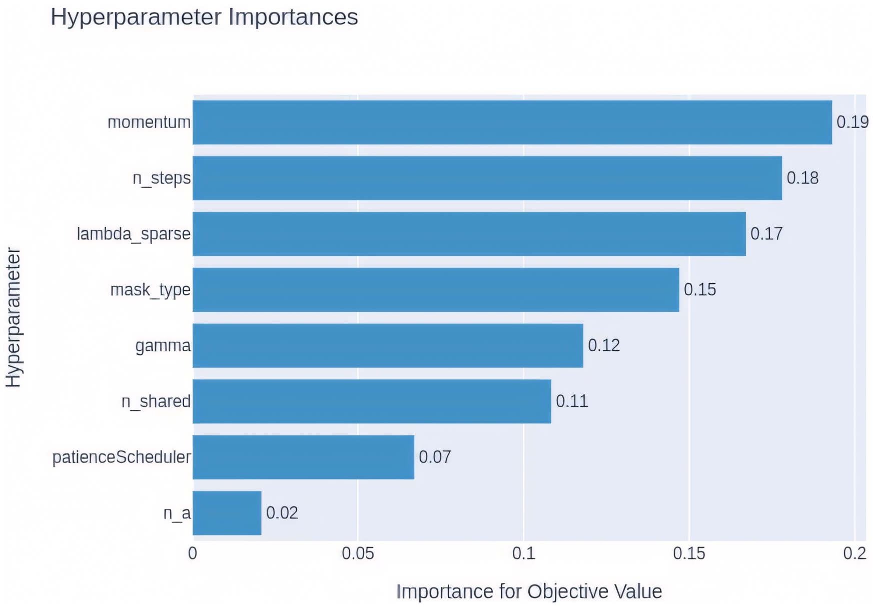


Figure 9.2: Optuna hyperparameter importance plot for TabNet trained on all LQ masses and tested on mass of 700 GeV, with LQ $\sigma = 1\,\text{pb}$.

It appears that the model tends to perform better with a higher number of steps and a higher value of gamma. This is expected since the model has 71 features to select from and understandably, some features provide more separation power than others. This notion is expanded upon in Section 10.

9.4.1 Evaluated TabNet Models

Using the information from Optuna, numerous models were trained and evaluated using the following parameters, descriptions of which can be found in Section 7.5.1.

- n_steps $\in$ {1,2,3}
- n_shared $\in$ {2,3}
- mask_type $\in$ {'entmax', 'sparsemax'}
- gamma = 1.4
- momentum = 0.1
- n_a = 64
- lambda_sparse = 2e-4

n_steps	n_shared	mask_type	Significance
1	3	entmax	1.373
2	3	entmax	1.275
3	3	entmax	1.400
3	2	entmax	1.265
3	3	sparsemax	1.167

Table 9.3: Significance (simplified) for different variations of TabNet models trained on all LQ masses, and tested on 500 GeV.

Overall, these TabNet models achieved a similar performance, except for when the sparsemax masking function was used.

9.5 XGBoost

Similarly to TabNet, XGBoost also uses a relatively high number of parameters and rather than attempting to estimate a good selection of parameters using trial and error, Optuna with pruning is used.

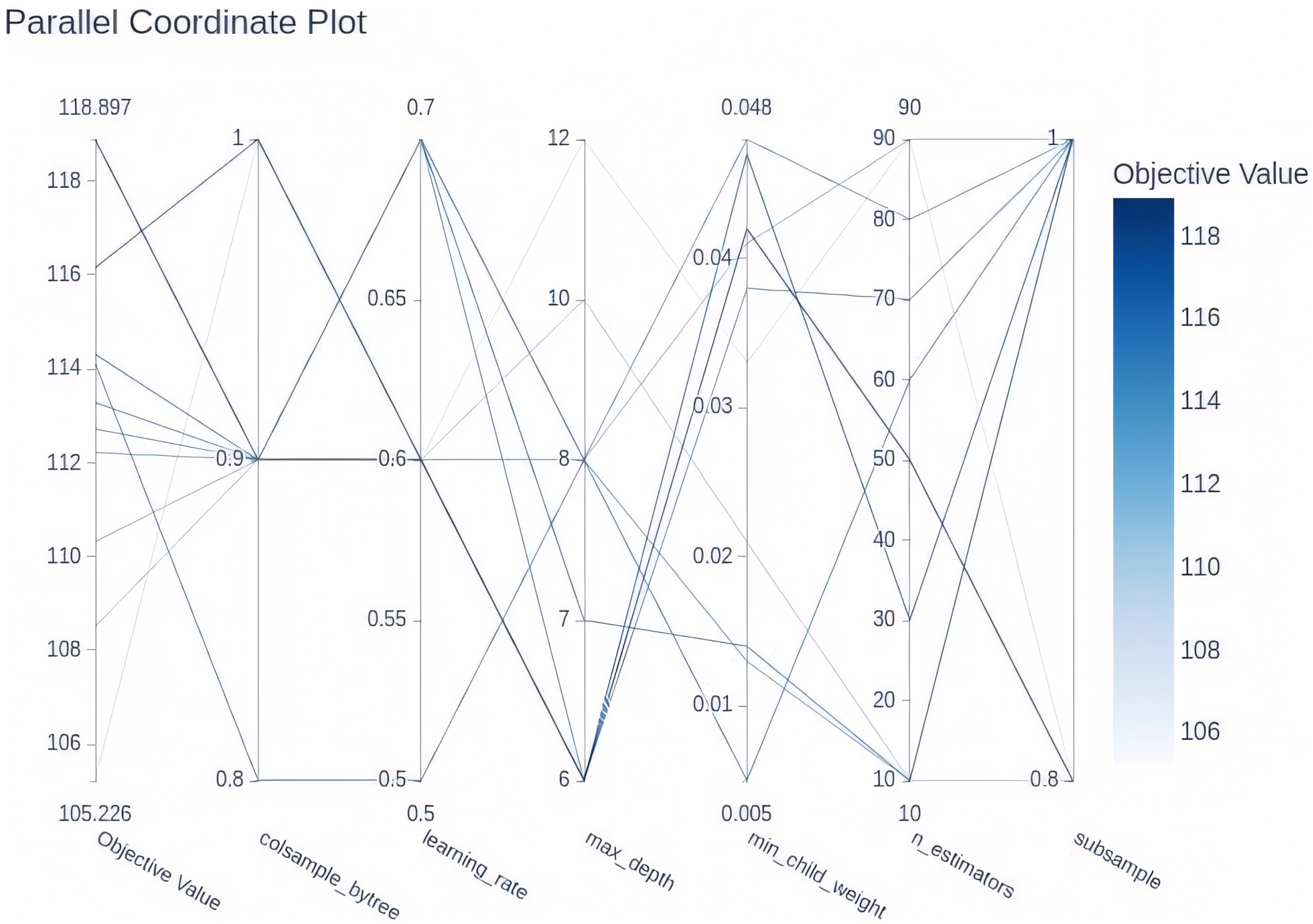


Figure 9.3: Optuna coordinate plot for XGBoost trained on all LQ masses and tested on mass of 700 GeV, with LQ $\sigma = 1\,\mathrm{pb}$.

The first striking feature of Figure 9.3 is the close similarity of the top objective values of TabNet and XGBoost, 118.3 and 118.9, respectively. Moreover, the best performing trial used `max_depth = 6` which suggests that an XGBoost decision tree with a smaller number splits tends to perform better.

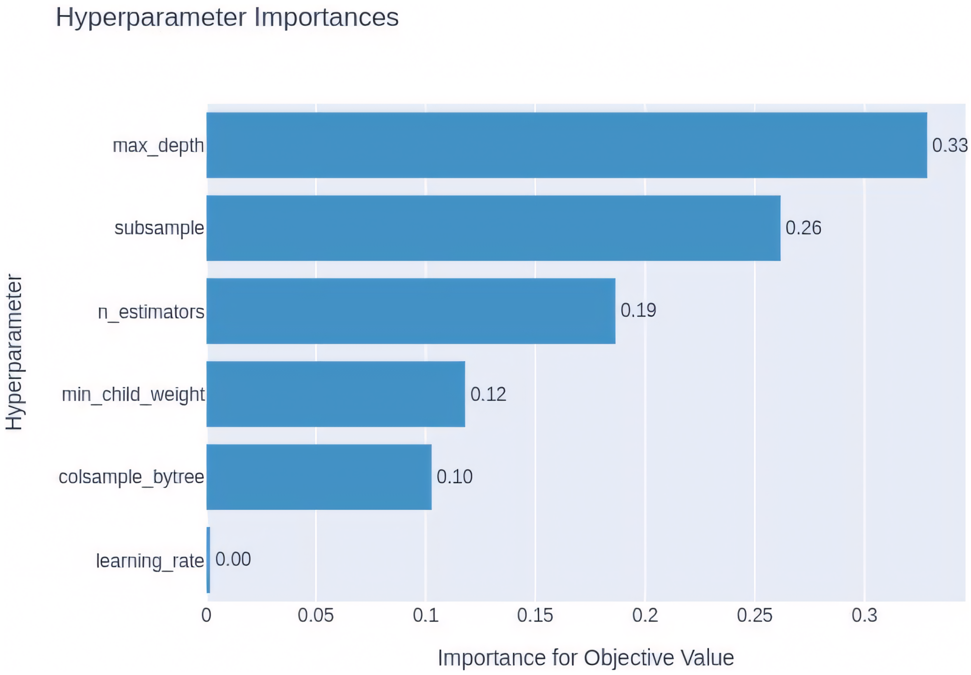


Figure 9.4: Optuna hyperparameter importance plot for XGBoost trained on all LQ masses and tested on mass of 700 GeV, with LQ $\sigma = 1\,\text{pb}$.

From Figure 9.4 it is evident that the value of `subsample` also holds high importance. This parameter allows for the use of bagging for $subsample \neq 1$. Seven out of the ten trials used `subsample = 1` and thus involved the entire training dataset. On the other hand, the best performing trial used `subsample = 0.8` which means that each learner only uses 80% of the training dataset.

9.5.1 Evaluated XGBoost Models

Thanks to the speed at which XGBoost learns, multiple models are trained and evaluated. The purpose of this analysis is to find a good combination of the number of trees and maximum tree depth. Other parameters are assigned values provided by Optuna and kept constant.

- max_depth $\in \{4, 6, 8, 10\}$
- n_estimators $\in \{15, 30, 50, 70, 90\}$
- learning_rate $= 0.6$
- min_child_weight $= 0.042$
- subsample $= 0.8$
- colsample_bytree $= 0.9$
- gamma $= 0$

n_estimators	max_depth	Significance
15	6	1.207
30	6	1.219
50	4	1.253
50	6	1.265
50	8	1.250
50	10	1.216
70	6	1.195
90	6	1.124

Table 9.4: Significance (simplified) for different variations of XGBoost models trained on all LQ masses, and tested on 500 GeV.

From the acquired XGBoost model performances, it appears that the maximum tree depth does not impact the performance as much as the number of estimators. While the results are consistent for lower number of estimators, when a higher number of estimators is used, the performance rapidly drops.

9.6 Selected Models

The best performing models on the LQ mass of 500 GeV are listed below.

1. **BNN**

   number of hidden layers = 2

   dimension of hidden layers = [20, 50], [50, 20]

   hidden layer activation function = ReLU

   dropout (after every hidden layer) = 0

   output layer function = LogSoftmax

2. **TabNet**

   `n_steps` = 3

   `n_shared` = 3

   `mask_type` = "entmax"

   `gamma` = 1.4

   `momentum` =0.1

   `n_a` = 64

   `lambda_sparse` = 2e-4

3. **XGBoost**

   `max_depth` = 6

   `n_estimators` = 50

   `learning_rate` = 0.6

   `min_child_weight` = 0.042

   `subsample` = 0.8

   `colsample_bytree` = 0.9

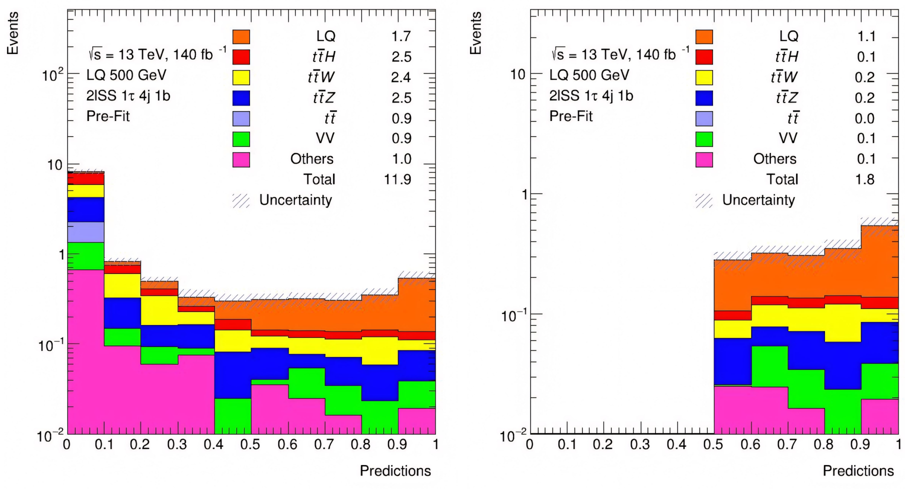
   `gamma` = 0

Out of all three models, the TabNet classifier achieved the highest significance with the XGBoost model in close second place. On the other hand, despite the Bayesian MLP performing better than its traditional counterpart, its performance wasn't on par with the other two models. For that reason, the focus of the analysis remains on TabNet and XGBoost only.

9.7 Optimal Threshold

When the model receives a test event to classify, it estimates seven probabilities where each value represents the probability that the given event belongs to the corresponding class. Since we are interested in the separation of the signal from background, the signal probability is analyzed.

The network output of the selected TabNet model before and after a threshold cut is shown in Figure 9.5. The optimal threshold corresponds to the probability at which the obtained significance is the highest. The approximation of significance and the amount of signal and background at different thresholds is shown in Figure 9.5. After the optimal threshold is found and applied to the network predictions in Figure 9.6, the significance improves from the default 0.53 to 1.40. The confusion matrices before and after the application of the selected threshold are given in Appendix C.



(a) Before threshold cut

(b) After threshold cut

Figure 9.5: Network output corresponding to the signal class before and after threshold cut for the selected TabNet model tested on mass of 500 GeV, with LQ $\sigma = 0.01\,\text{pb}$.

Figure 9.7 shows the ROC curves of all seven processes. While the LQ and VV processes are classified relatively well, as shown by their relatively high AUC, the other processes are not. Most notably, $t\bar{t}$ performs very poorly, which is due to the small number of events in the test dataset which is also apparent from its jugged ROC curve.

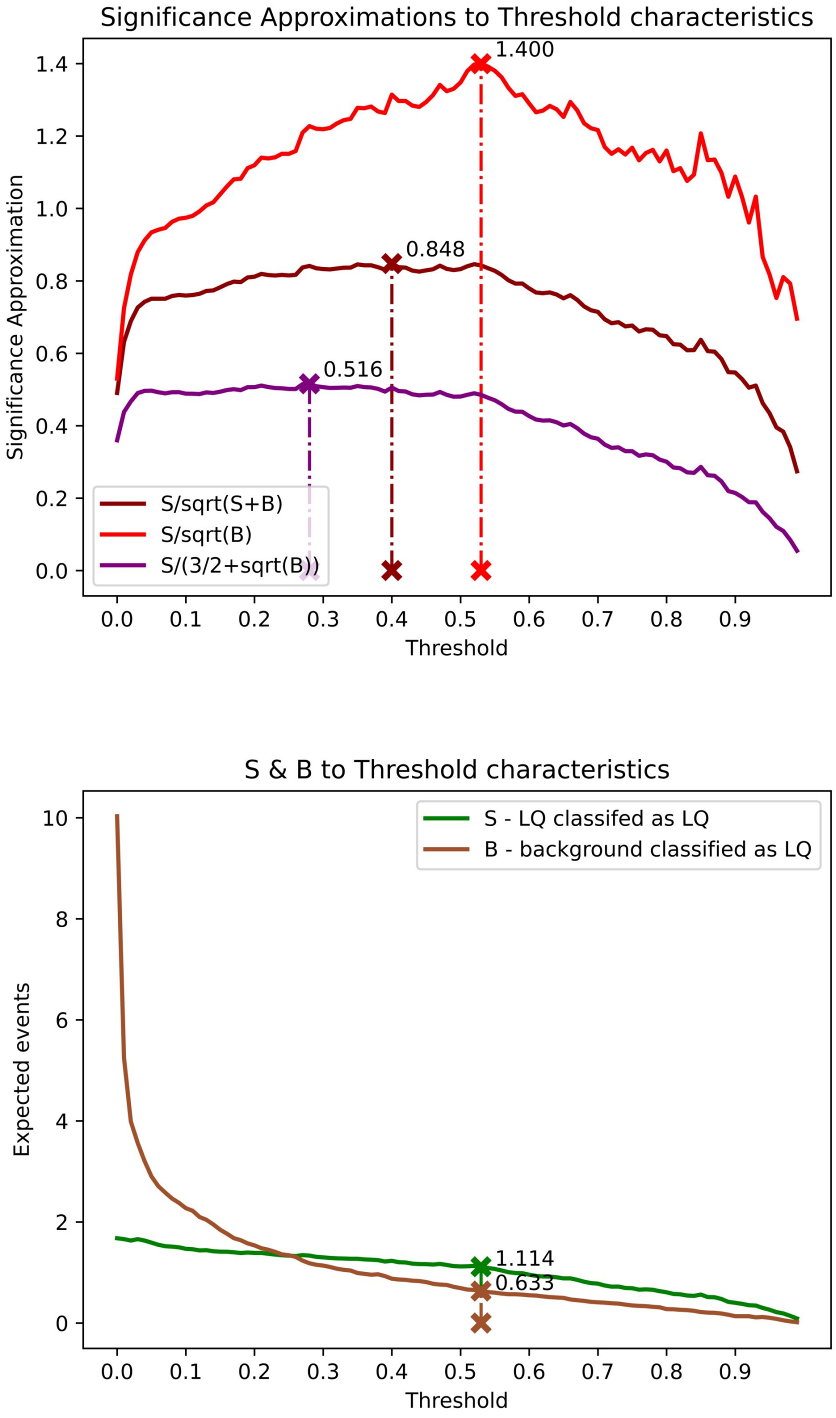


Figure 9.6: Significance approximation per threshold and signal to background ratio for selected TabNet model tested on mass of 500 GeV, with LQ $\sigma = 0.01\,\mathrm{pb}$.

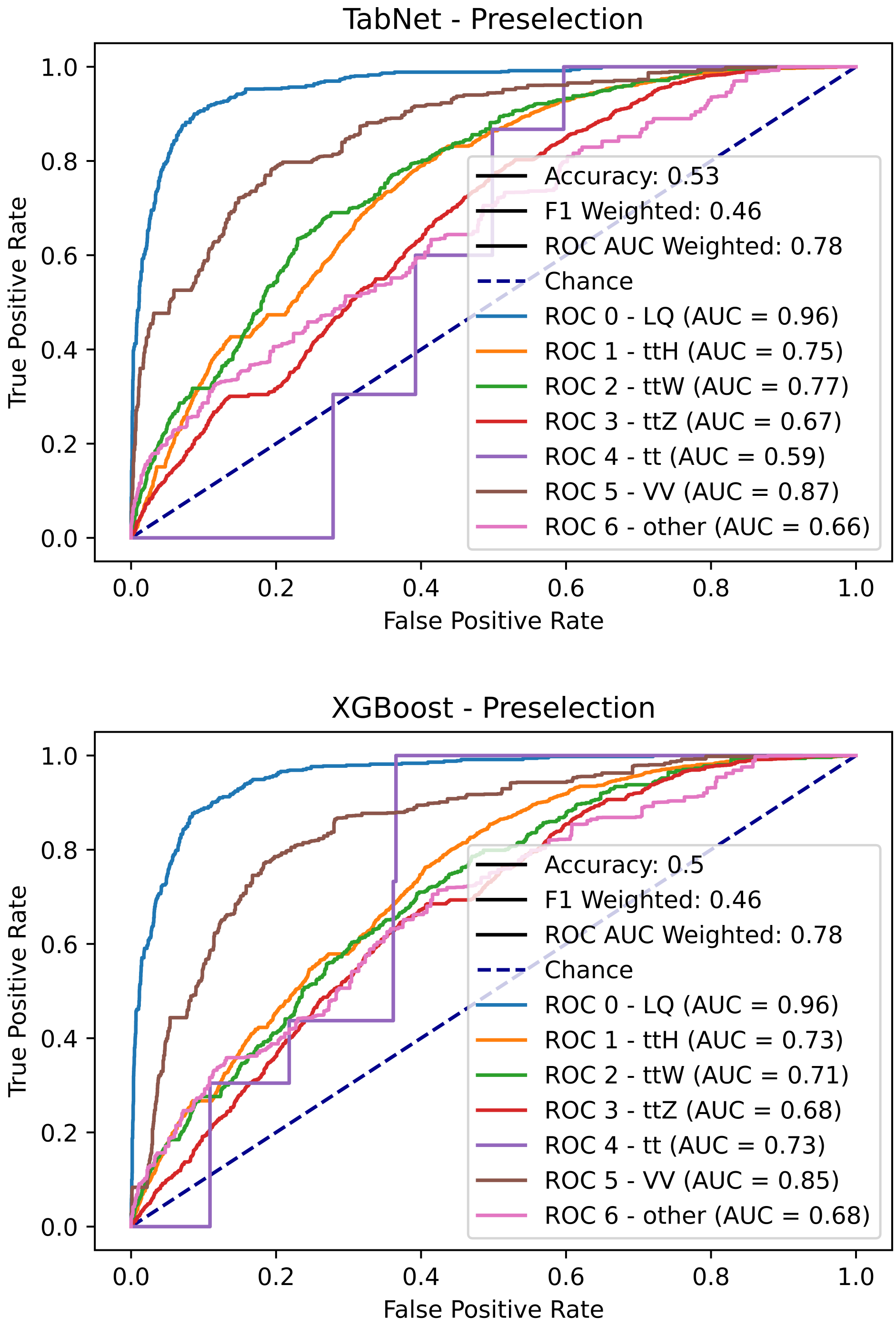


Figure 9.7: Weighted ROC curves for the selected TabNet model (upper) and XGBoost model (lower), for combined masses of LQ, with LQ $\sigma = 0.001\,\text{pb}$.

Chapter 10

Feature Importance

10.1 Feature Ranking

It is important to point out that due to the randomness of the sampling process, the constructed training and test datasets differ with each sampling run. While the raw number of samples per class stays the same, the individual events do not, and therefore the results of the training and test procedures may differ. One aspect that may differ is the feature importance constructed by a model that uses the sampled data.

The selected models were run 20 times and the feature importance of all 71 features was noted for each iteration. The 20 most important features and their importance for each model is shown in Figure 10.1.

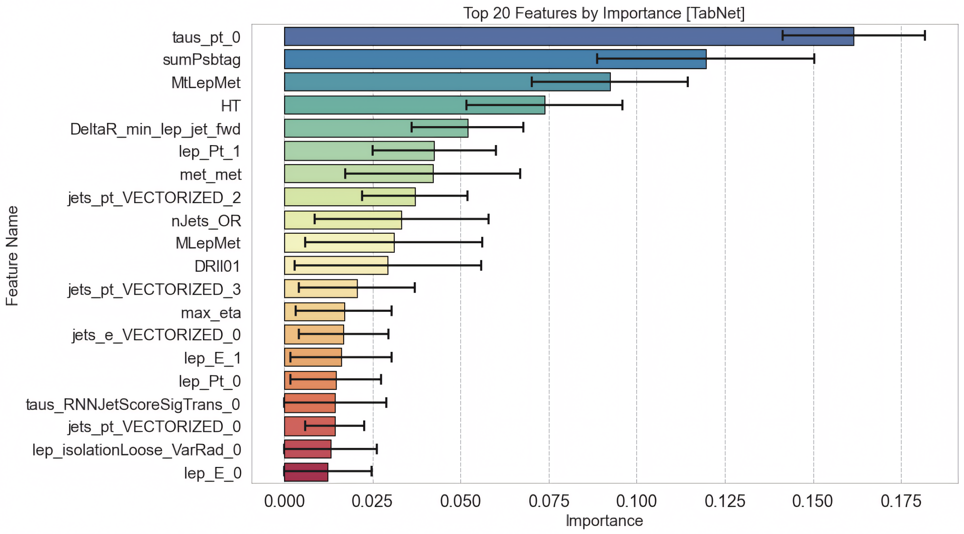


Figure 10.1: Feature importance of top 20 features of the selected TabNet model trained on all features, using all LQ masses, computed as averages of 20 trials.

Figure 10.1 shows the idea that the TabNet model focuses on a small number of features. In fact the first four listed features have a higher combined mean importance than the 16 remaining features. This is interesting because it means that the model bases its separation on a very small subset of the available features. This is even more extreme in the case of the XGBoost model as illustrated in Figure 10.2.

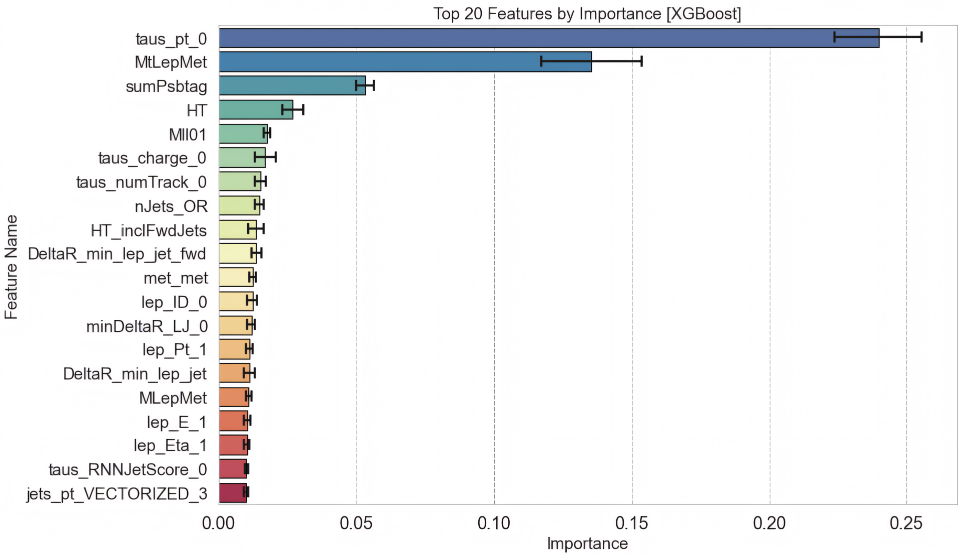


Figure 10.2: Feature importance of top 20 features of the selected XGBoost model trained on all features, using all LQ masses, computed as averages of 20 trials.

While both models have the top four features in common, the XGBoost model has a higher disparity between its top and bottom features. On the other hand, the standard deviation of the feature importance belonging to the XGBoost model is relatively low for all shown features compared to the TabNet model. It can be inferred that the training process of the TabNet model exhibits higher variability, whereas the training process of XGBoost appears to be more consistent.

The list of feature importance for the top 60 features of both models are given in Appendix D.

Moreover, a comparison can be made with the previous analysis [11]. Out of the top 20 features, eight are common to both analyses, and out of the top six features, five are the same. Feature "HT" which is ranked in fourth place cannot be found in the V6 feature importance list, but otherwise it can be said that features in V6 and V8 of the dataset hold similar importance for the TabNet architecture.

Figure 10.3 shows the distributions of the four most important features. Note that the plots show the distributions for all 12 LQ masses combined (500 - 1600 GeV). This must be considered as the distributions vary from mass to mass. An example with a distribution of "taus_pt_0" and "MtLepMet" for LQ mass of 500 GeV and 1600 GeV is given in Appendix E.

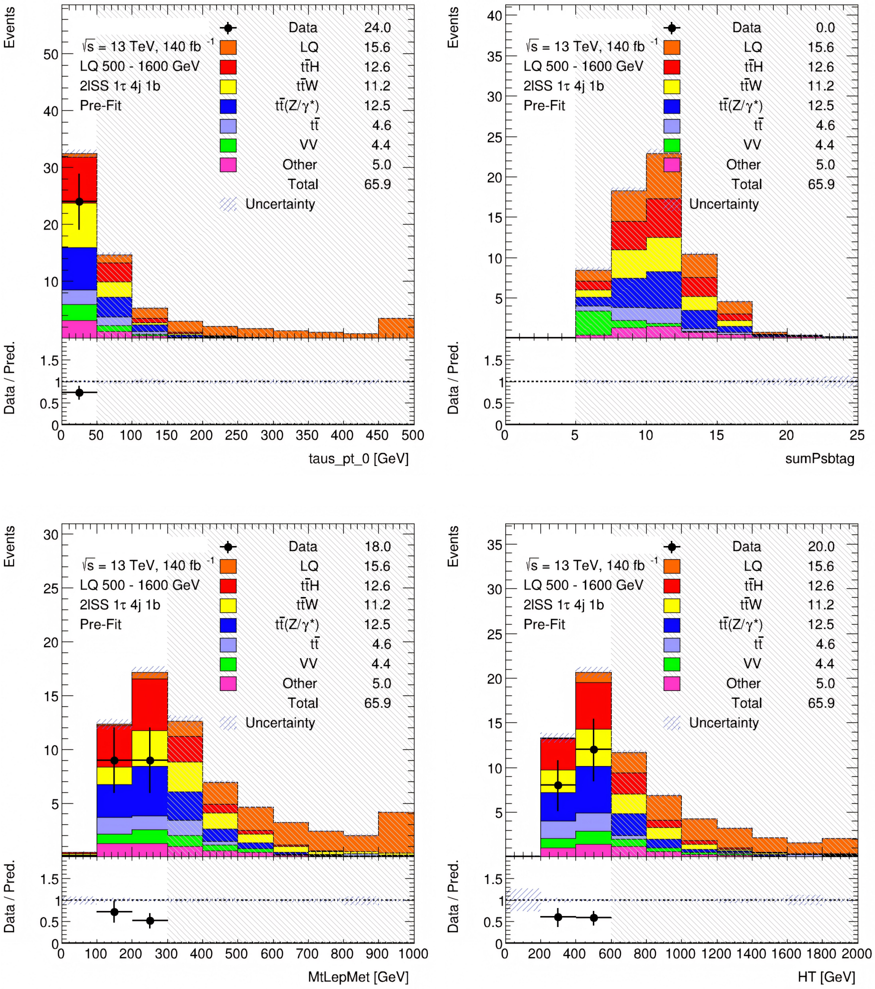


Figure 10.3: Histograms with distributions of the top four features, with LQ $\sigma = 0.001$ pb.

The black dots in Figure 10.3 correspond to data from the experiments at the LHC. From these we may see how closely the distributions of the simulated data resemble the recorded data. Any deviations between the two are also shown in the histogram below the main distribution plot. Note that these include statistical uncertainty.

Some bins in Figure 10.3 are hashed and do not contain any data points. This is called "blinding" and it is controlled by the `BlindingThreshold` parameter which is set to 0.1 for these plots. In each bin there is a ratio of signal to background, and the blinding threshold sets the maximum allowed ratio of signal to background for data to be displayed. Therefore, bins with a ratio greater than the blinding threshold are blinded and appear hashed.

10.2 Feature Correlation

The correlation of features must also be considered. Figure 10.4 shows that MtLepMet and HT are positively correlated with most other features. Notably, MtLepMet has a very strong correlation with MLepMet (0.91) and relatively strong correlation with met_met (0.78), HT (0.74), lep_Pt_1 (0.7). The other striking feature is DeltaR_min_lep_jet_fwd that is negatively correlated with all other features in the top ten.

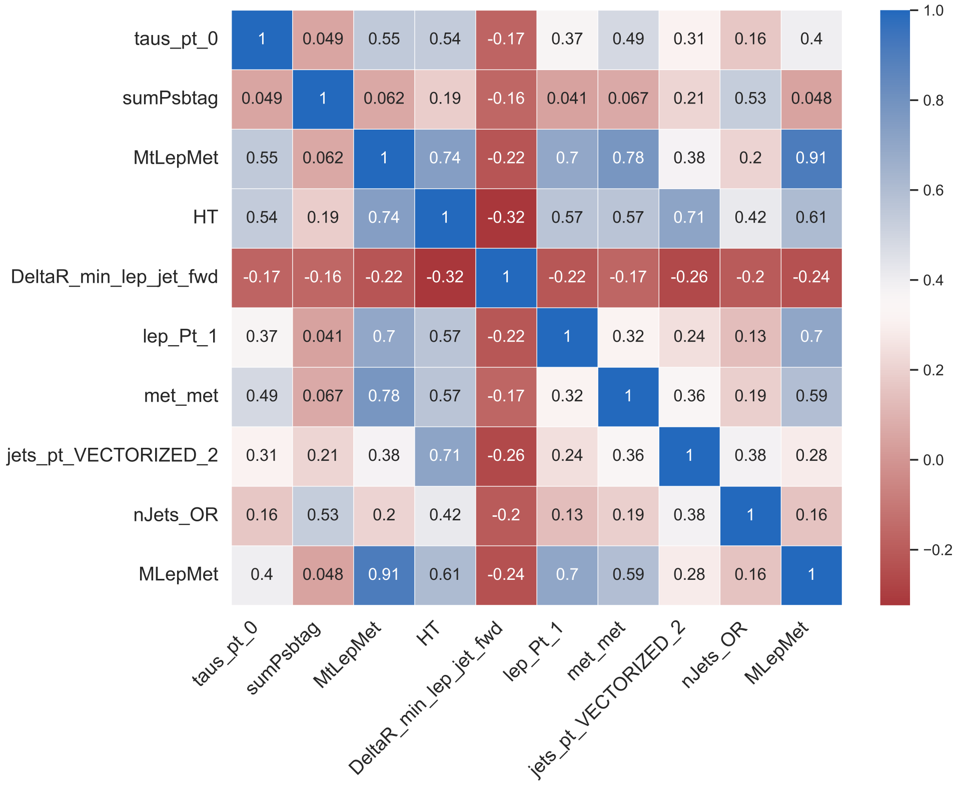


Figure 10.4: Pearson correlation coefficients for the 10 most important features.

Chapter 11

Efficient Models

Using the obtained list of important features, a new dataset was generated containing only the 20 most important features. When the preparation of this dataset was being done, a new idea had emerged which was that instead of limiting the training to events corresponding to the preselection cut, we could train a model on all available events which could provide a couple of benefits:

- Higher number of events for training as shown in Table 11.1.
- Better representation of under-represented classes.
- Improved representation of each class in the feature space.

Process	With preselection	Without preselection
LQ (all masses)	20,167	158,528
$t\bar{t}H$	15,377	724,120
$t\bar{t}W$	2,775	532,203
$t\bar{t}Z$	8,432	1,453,565
$t\bar{t}$	20	159,994
VV	1,897	3,395,026
"Other"	1,813	302,473

Table 11.1: Recorded numbers of raw events with and without preselection, where events with negative weights were removed.

Preselection conditions corresponding to the 2lSS+1τ channel are given in Appendix A.

While this approach has apparent advantages, a disadvantage could be that the simulations without preselection are not validated against the recorded data.

11.1 Training on Events Without Preselection

Using the 20 most important features, a new model was constructed for each of the selected algorithms. For training of the model, the entire dataset corresponding to events without preselection was used. For testing, only events corresponding to the preselection cut were used.

11.1.1 Improved Models

The training process of the TabNet model had taken a little over 10 hours as it converged very slowly. However, if we were to use all 71 features, this would have taken much longer. The resultant ROC curves for the signal and backgrounds are shown in Figures 11.1 and 11.2.

The performance of the XGBoost model very closely resembled the performance of the TabNet model. However, the TabNet model achieved a slightly better AUC for all backgrounds except the diboson. Especially the classification of the $t\bar{t}$ samples improved in both models. Due to having a larger dataset for testing, the smoothness of the curves for all seven processes also improved. However, it is important to note that an improvement in smoothness does not necessarily signify an improvement in performance.

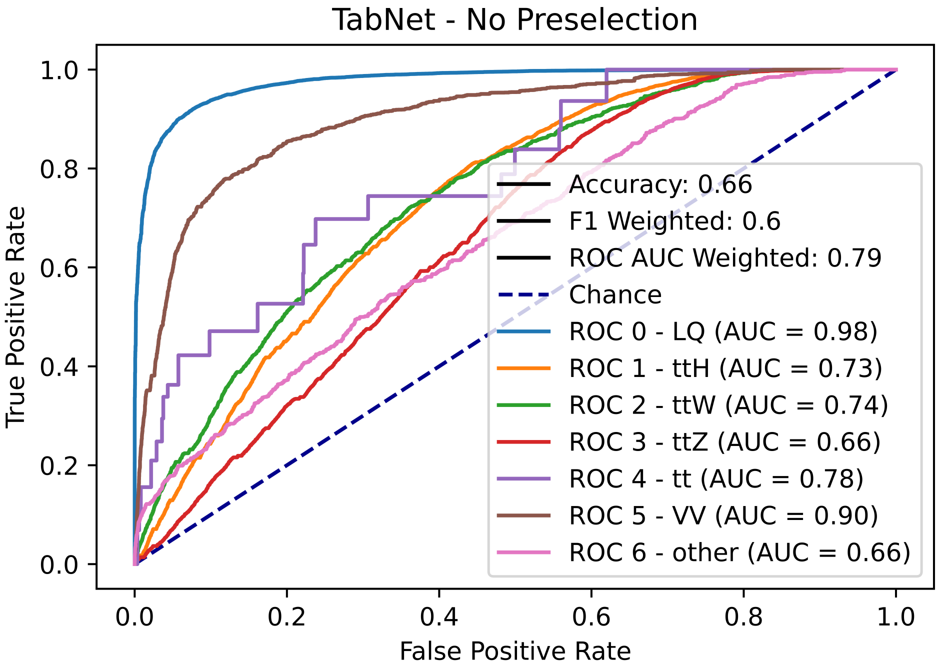


Figure 11.1: Weighted ROC curves of the TabNet model trained on all events without preselection and tested on events with preselection, for combined masses of LQ, with LQ $\sigma = 0.001\,\text{pb}$.

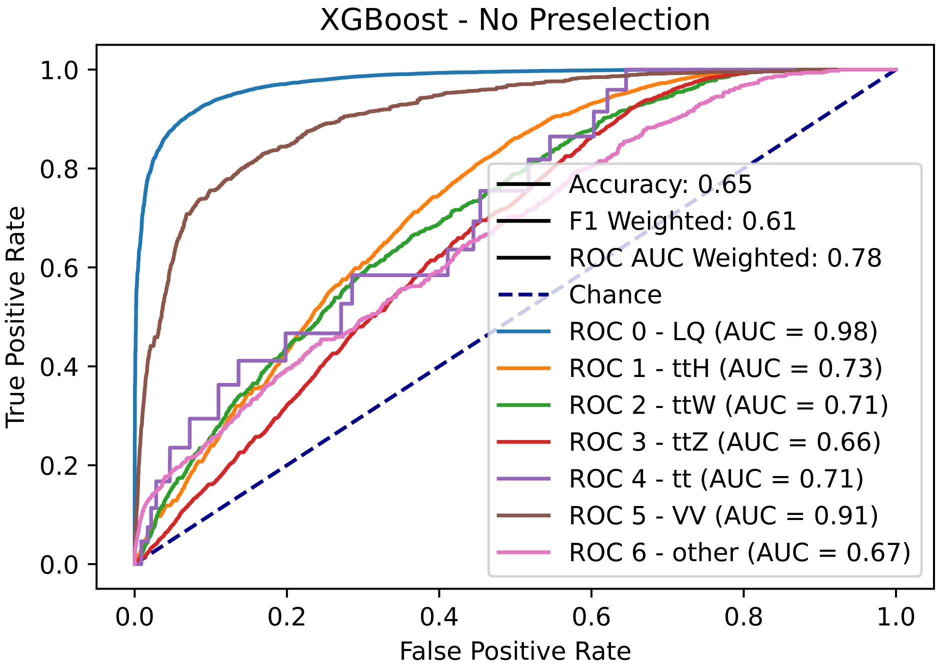


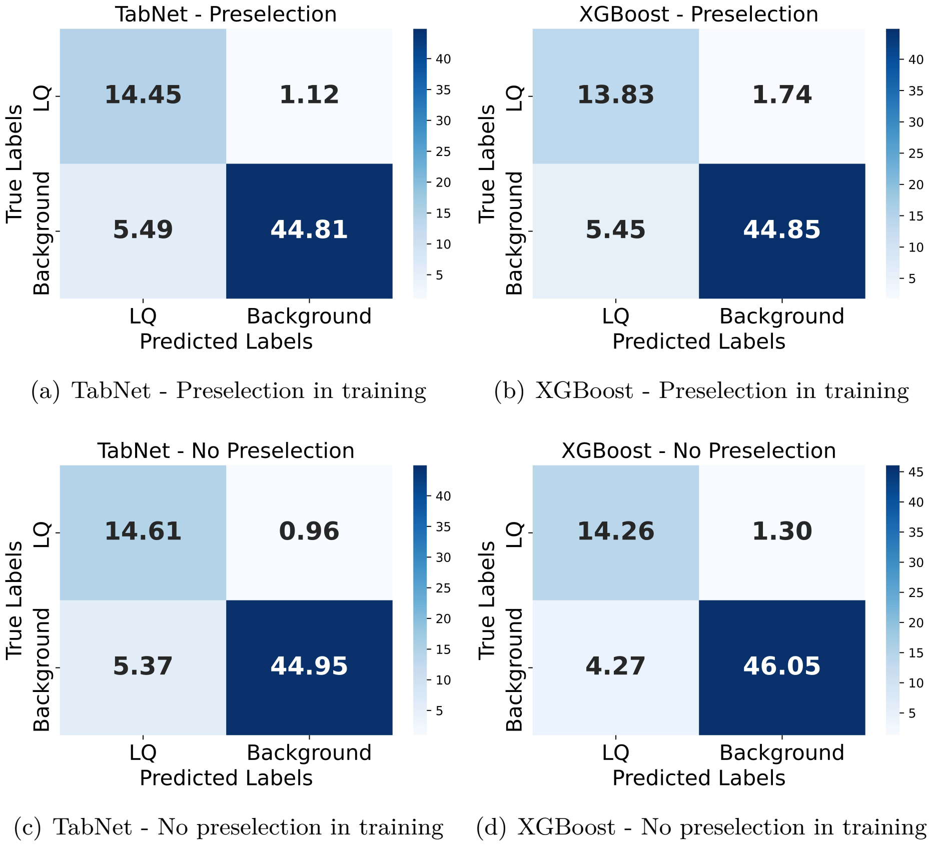
Figure 11.2: Weighted ROC curves of the XGBoost model trained on all events without preselection and tested on events with preselection, for combined masses of LQ, with LQ $\sigma = 0.001\,\text{pb}$.

Performance measure	TabNet before	TabNet after	XGBoost before	XGBoost after
Accuracy	0.53	0.66	0.50	0.65
F1 Score	0.46	0.60	0.46	0.61
Combined AUC	0.78	0.79	0.78	0.78
LQ AUC	0.96	0.98	0.96	0.98
$t\bar{t}$H AUC	0.75	0.73	0.73	0.73
$t\bar{t}$W AUC	0.77	0.74	0.71	0.71
$t\bar{t}$Z AUC	0.67	0.66	0.68	0.66
$t\bar{t}$ AUC	0.59	0.78	0.73	0.71
VV AUC	0.87	0.90	0.85	0.91
"Other" AUC	0.66	0.66	0.68	0.67

Table 11.2: Comparison of accuracy, F1 score and AUC of models trained on preselected events only (before) and models trained on events without preselection (after). The displayed values correspond to Figures 9.7, 11.1 and 11.2.

All four models have a combined AUC of around 0.78. However, the models trained on data without preselection achieve a better accuracy and F1 score as shown in Table 11.2. This can be attributed to the models' improved sensitivity to the background processes. Notably, classification $t\bar{t}$ events had improved quite significantly in the case of the TabNet model which is due to the model having a lot more events to train on. Lastly, the AUC for the LQ process has increased very little, from 0.96 to 0.98 for both TabNet and XGBoost which is not a significant improvement.

In order to see if any significant improvement had been made from the models trained on preselected data and models trained on data outside of preselection, confusion matrices of all four models are shown in Figure 11.3. The weights of the test sets for the models trained on preselected data are scaled up by five, to correspond with the original yields. Note that that while the upper-left field shows the TP, the lower-right field does not show the TN. Therefore these matrices focus on showing TP, FP, and FN for the LQ events.



(a) TabNet - Preselection in training

(b) XGBoost - Preselection in training

(c) TabNet - No preselection in training

(d) XGBoost - No preselection in training

Figure 11.3: Confusion matrices for models trained on preselected events only (upper) and models trained on events without preselection (lower). Tested on all LQ masses in preselection, with LQ $\sigma = 0.001\,\text{pb}$.

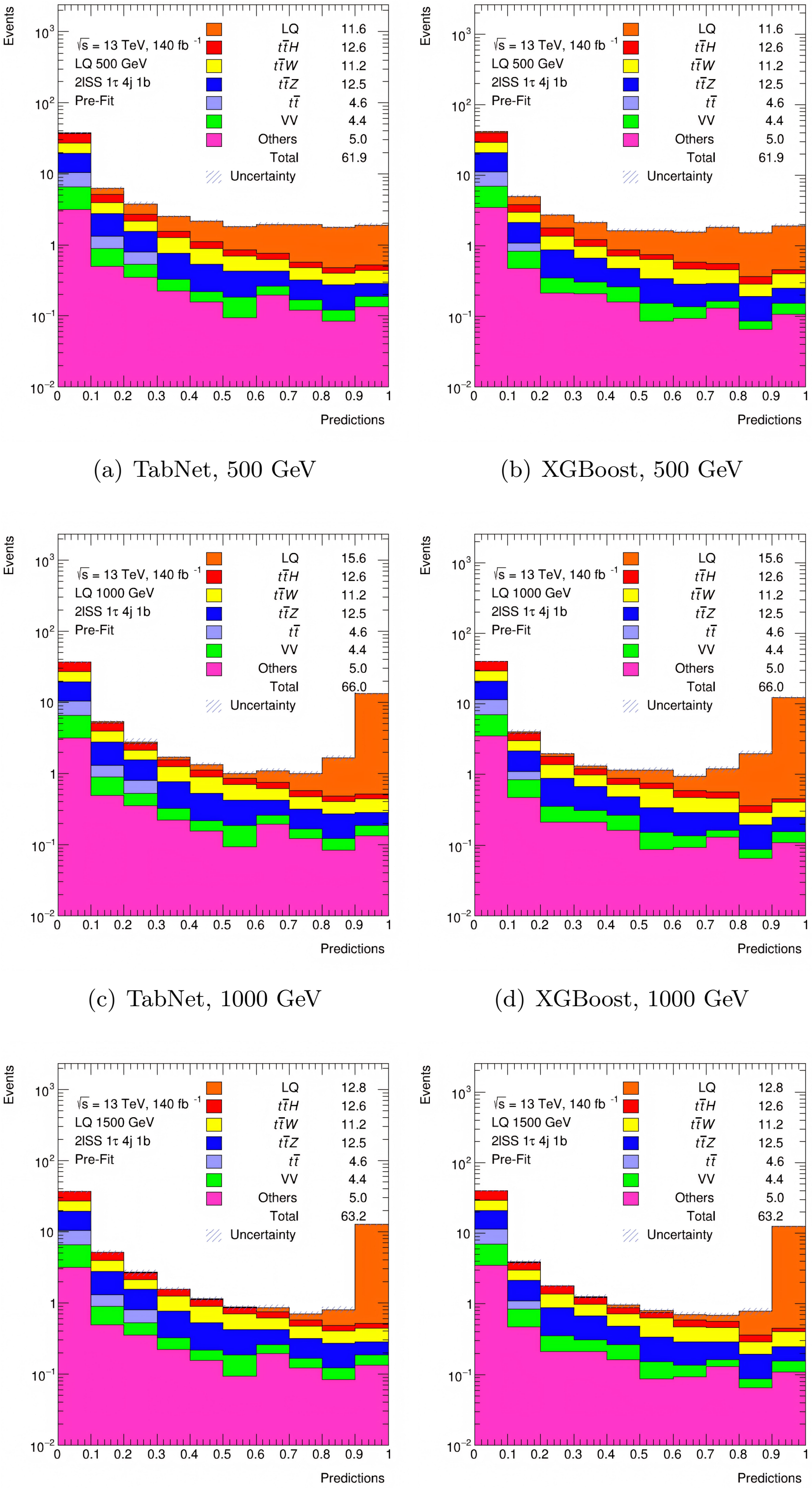
While for TabNet there is only a small increase in TP and a decrease in FP and FN, XGBoost shows a more significant change of TP from 13.83 to 14.26 and a decrease in FP from 5.45 to 4.27. This suggest that a greater amount of signal events are classified as signal and a greater amount of background is classified as background. Therefore, this increase in sensitivity and specificity suggests an improvement in separation of signal from background.

11.1.2 Network Output

There is an apparent trend that the models struggle with the separation of lower LQ masses than the larger masses, as illustrated in Figure 11.4. Furthermore, from the plots we can observe that as the mass of the signal increases, so does the separation. Both networks struggle with separation of the 500 GeV signal. This improves for the signal events corresponding to 600 GeV, and for masses above 700 GeV, the separation improves less rapidly. Most of all, despite the differences of the underlying machine learning algorithms, both the TabNet and XGBoost models obtain very similar network output distributions.

The worse separation of LQ signal on lower masses is likely due to the fact that the feature distributions of lower LQ masses resemble the feature distributions of the background more closely, than larger LQ masses. This can be seen in the plots in Appendix E. As a result, for large masses of LQ, the classifiers can select a hyperplane to separate the signal from the background more easily, as the underlying distributions differ more significantly than in lower masses. The difference in separation of signal from background between LQ masses can be seen in Figure 11.4.

The network output corresponding to the signal class for all masses and both models is given in Appendix F.



(a) TabNet, 500 GeV

(b) XGBoost, 500 GeV

(c) TabNet, 1000 GeV

(d) XGBoost, 1000 GeV

(e) TabNet, 1500 GeV

(f) XGBoost, 1500 GeV

Figure 11.4: Network output corresponding to the signal class for three different test masses (500, 1000, 1500 GeV), with LQ $\sigma = 0.01$ pb.

Chapter 12

Binary Classifier

After seeing the trained models struggle with the classification of background, one more idea arose, and that was to train a binary classifier and test its performance against the previous models. A binary classifier is a machine learning algorithm which categorizes data into two classes - signal and background. Unlike in a multi-class classifier, for an event classified as background, the classifier does not predict which class the event belongs to. Its focus is solely on the separation of one class from the other.

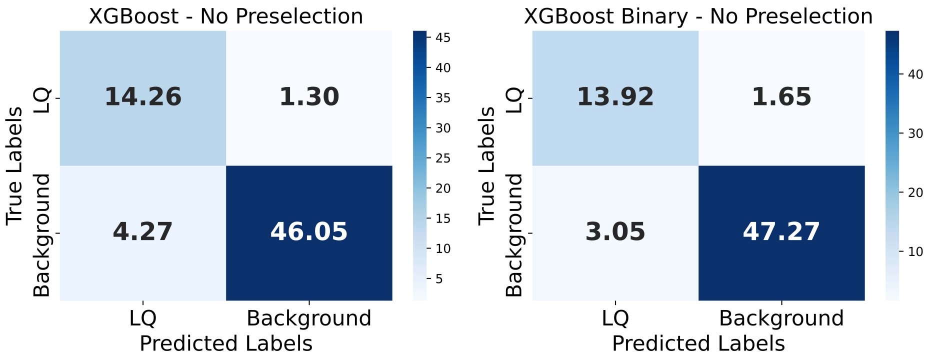
Using the hyperparameters from Chapter 9, multiple XGBoost binary classifiers were trained on data without preselection, evaluated on data with preselection, and their significance values at the optimal threshold were calculated. Different combinations of the following parameters were used:

- max_depth $\in \{4, 6, 8\}$
- n_estimators $\in \{30, 50, 70\}$
- learning_rate $= 0.6$
- min_child_weight $= 0.042$
- subsample $\in \{0.8, 1\}$
- colsample_bytree $= \in \{0.9, 1\}$
- gamma $= 0$

n_estimators	max_depth	subsample	colsample	Significance
30	6	0.8	0.9	1.916
50	4	0.8	0.9	1.912
50	6	0.8	0.9	1.949
50	6	1	0.9	1.955
50	6	0.8	1	1.903
50	8	0.8	0.9	1.880
70	6	0.8	0.9	1.935

Table 12.1: Significance (simplified) for different variations of XGBoost binary classifiers trained on data without preselection, and tested on data with preselection with all LQ masses as signal, LQ $\sigma = 0.001$ pb.

The model with the highest achieved significance of 1.955 consists of 50 estimators, has a maximum depth of 6, and does not use subsampling. In order to compare its performance against the previous XGBoost model, confusion matrices are constructed as shown in Figure 12.1.



(a) XGBoost - No preselection in training (b) XGBoost Binary - No preselection in training

Figure 12.1: Confusion matrices for XGBoost multi-class classifier (left) and binary classifier (right) trained on preselected events only. Tested on all LQ masses in preselection, with LQ $\sigma = 0.001$ pb.

Figure 12.1 shows that the obtained sensitivity is lower in the binary classifier than in the multi-class classifier. On the other hand, there is an increase in specificity as the number of TN increased from 46.05 to 47.27. However, the overall performance of the binary classifier does not surpass the previous model. The plot of the expected upper limit at 95% CL of Leptoquark cross-section using the network output of the binary classifier is given in Appendix G.

In the following chapter, Chapter 13, it is shown that the multi-class models trained on data without preselection achieve the best expected upper limit at 95% CL of Leptoquark cross-section out of all models analyzed in this thesis.

Chapter 13

Results

13.1 Expected Limits

The last step of the analysis is to determine the expected upper limit of cross-section of the LQ pair production. This can either be done numerically by scaling the signal such that the resultant significance corresponds to 2σ which in turn corresponds to the 95% confidence level, or using TRExFitter that analyzes the network output and computes the expected limits automatically.

The expected upper limit of cross-section can be calculated as:

$$\sigma_e = \sigma_t \cdot \xi, \tag{13.1}$$

where σ_e is the expected upper limit of cross-section at a 95% confidence level, σ_t is the theoretical cross-section and ξ is the scaling factor applied to the signal.

13.1.1 Employment of TRExFitter

The process can be computed using TRExFitter. TRExFitter works with ROOT files (as described in Section 6.1), first ROOT files for each of the 12 masses are created. Each ROOT file needs to contain a tree which is populated by three branches, each containing different information:

- probability output of the network
- event weights - to scale the contribution of each event to the network output
- truth labels - to mark which events correspond to which process

Before running the TRExFitter configuration file for each mass, a replacement file that contains the ROOT file paths must be prepared. An example of a simplified configuration file is given in Appendix H. In order for the results to be accurate, it is important that each process has the correct yield. Therefore, for a test set which has a yield of 20% compared to the original dataset, the weights of the events must be scaled by five to correspond to the original yields.

When the configuration file is executed, TRExFitter produces histograms out of the given ROOT files, which it then uses for the computation of the limits. The asymptotic `LimitType` was used to produce the results in Tables 13.1, 13.2, 13.3 and 13.4. Statistical uncertainty is also computed, giving the 68% and 95% confidence intervals.

Mass [GeV]	**500**	**600**	**700**	**800**	**900**	**1000**
Limit [pb]	0.006671	0.004568	0.002981	0.002524	0.002418	0.002076
Limit + σ [pb]	0.009756	0.006756	0.004460	0.003798	0.003659	0.003166
Limit + 2σ [pb]	0.014270	0.01009	0.006877	0.005958	0.005928	0.005137
Limit - σ [pb]	0.004807	0.003292	0.002148	0.001818	0.001742	0.001496
Limit - 2σ [pb]	0.003580	0.002452	0.001600	0.001355	0.001298	0.001114

Table 13.1: Asymptotic expected upper limit at 95% CL on cross-section in pb, with 68% and 95% confidence intervals, obtained with the TabNet model.

Mass [GeV]	**1100**	**1200**	**1300**	**1400**	**1500**	**1600**
Limit [pb]	0.002044	0.002040	0.001780	0.001972	0.001970	0.002047
Limit + σ [pb]	0.003107	0.003099	0.002721	0.003016	0.003010	0.003129
Limit + 2σ [pb]	0.005193	0.005255	0.004644	0.005196	0.005203	0.005433
Limit - σ [pb]	0.001473	0.001470	0.001282	0.001421	0.001420	0.001475
Limit - 2σ [pb]	0.001097	0.001095	0.000955	0.001058	0.001057	0.001099

Table 13.2: Asymptotic expected upper limit at 95% CL on cross-section in pb, with 68% and 95% confidence intervals, obtained with the TabNet model.

Mass [GeV]	**500**	**600**	**700**	**800**	**900**	**1000**
Limit [pb]	0.006535	0.004417	0.003221	0.002517	0.002267	0.002071
Limit + σ [pb]	0.009586	0.006542	0.004829	0.003791	0.003439	0.003154
Limit + 2σ [pb]	0.01410	0.009859	0.007476	0.005997	0.005535	0.005159
Limit - σ [pb]	0.004709	0.003182	0.002321	0.001814	0.001634	0.001492
Limit - 2σ [pb]	0.003508	0.002371	0.001729	0.001351	0.001217	0.001112

Table 13.3: Asymptotic expected upper limit at 95% CL on cross-section in pb, with 68% and 95% confidence intervals, obtained with the XGBoost model.

Mass [GeV]	**1100**	**1200**	**1300**	**1400**	**1500**	**1600**
Limit [pb]	0.002098	0.002083	0.001847	0.002053	0.001993	0.002112
Limit + σ [pb]	0.003201	0.003193	0.002823	0.003149	0.003074	0.003256
Limit + 2σ [pb]	0.005266	0.005293	0.004728	0.005287	0.005184	0.005487
Limit - σ [pb]	0.001512	0.001501	0.001331	0.001480	0.001436	0.001522
Limit - 2σ [pb]	0.001126	0.001118	0.000991	0.001102	0.001070	0.001134

Table 13.4: Asymptotic expected upper limit at 95% CL on cross-section in pb, with 68% and 95% confidence intervals, obtained with the XGBoost model.

There is a small but notable difference between the limits obtained with the TabNet model and the XGBoost model. While XGBoost achieves slightly lower expected limits for lower masses, such as 0.006535 pb compared to 0.006671 pb for 500 GeV as shown in Tables 13.1 and 13.3, TabNet achieves a lower limit for 1300 - 1600 GeV which is shown in Tables 13.2 and 13.4. Besides these minor differences, the obtained limits are quite similar.

13.2 Comparison of Expected Upper Limit of Cross-section

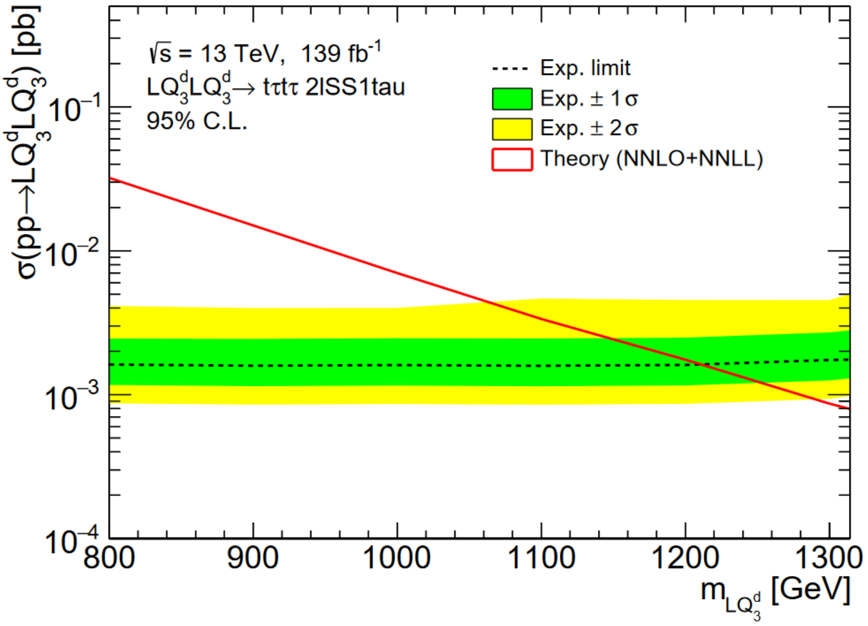
Figure 13.1 shows the comparison of the obtained expected upper limit of this and the previous analysis [11].

Note that for lower masses the new obtained limit is higher than the previous one which indicates weaker separation of signal from background on that mass range. It has been shown in this analysis, that the nature of a model trained on all masses is that it is able to separate larger masses well but struggles with lower masses, which explains the down-sloping shape of the curve. This is the case with limit results from both models. While the previous resulting limit is relatively flat, in the new analysis, the trained models achieve a significantly weaker sensitivity for lower masses. On the other hand, our expected upper limit for larger masses, specifically around 1300 GeV, is very similar to the previous result.

One possible explanation for the weaker separation results on lower masses of signal could be the difference between the yield of V8 and V6 data. Moreover, this difference is especially noticeable on lower masses. This may be due to the fact that the yields of all LQ masses in the V8 dataset are lower than the yields in the V6 dataset. For example, for 1600 GeV, the V8 dataset has a yield of 1215.6 compared to 1359.2 in the V6 dataset for LQ $\sigma = 1\,\mathrm{pb}$. Therefore, having fewer events to train from could likely affect the learning process of the TabNet and XGBoost models.

The expected upper limit plot obtained using XGBoost classification results, along with plots of both limits with linear scaling are given in Appendix I.

The LQ masses below 1180 GeV are expected to be excluded at 95% CL for the model parameters [4]. This is very close to the value obtained in the previous analysis [11]. This result can be improved by including systematic uncertainties which would reduce the sensitivity and thus reduce the expected mass limit.



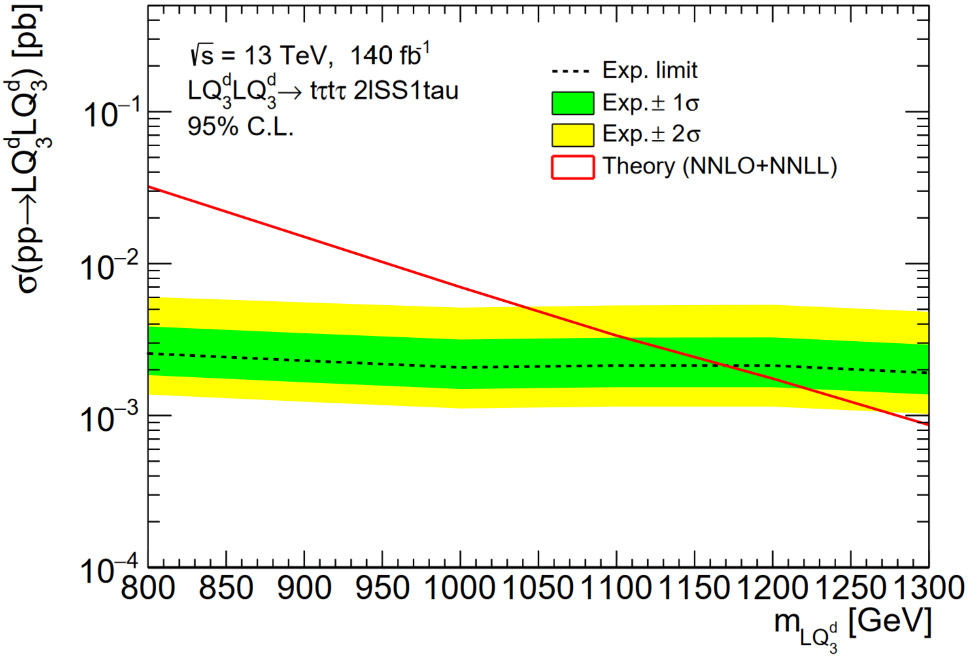


Figure 13.1: Comparison of limit obtained by previous result [11] (upper) and limit obtained during by this analysis (lower) using the TabNet model.

Chapter 14

Conclusion

The main objective of this thesis was to optimize several machine learning algorithms to effectively distinguish simulated events associated with the pair-production of Leptoquarks from various background processes.

After becoming familiar with the previous analysis and the technologies employed in high-energy physics analyses, we studied, trained, and evaluated four algorithms, namely TabNet, XGBoost, MLP, and Bayesian MLP. Among these, TabNet and XGBoost were selected for further analysis.

In this paper, the differences between the V6 and V8 datasets were examined, as well as any differences in the analyses themselves. This included assessing feature importance for each model. Additionally, a plot illustrating Pearson coefficients of correlation for the top ten important features was generated to identify any relationships among the most important features.

Using the gathered information, new, more efficient models were trained on events without preselection, including a binary classifier. The classification results of these models were utilized to estimate the upper limit of cross-section at a 95% confidence level for the LQ masses, ranging from 500 to 1600 GeV. The obtained limits were plotted and compared against the previous results.

In contrast to the previous limit [11], the two produced limits show a slightly different trend, with lower masses having a higher expected limit compared to larger masses. This difference is attributed to the models struggling with signal separation at lower masses. One possible explanation for this behavior could be the smaller size of the V8 dataset compare to the V6 dataset, but also the fact that the feature distributions of the lower LQ masses resemble the feature distributions of the background processes more closely than the large LQ masses.

Overall, the objectives of this thesis were successfully accomplished, and the content presented here will contribute to the foundation of ideas and methods employed in future Leptoquark searches which also include the effect of systematic uncertainties.

Bibliography

[1] Giordon Stark. *The Large Hadron Collider and the ATLAS Detector*. Springer International Publishing, Cham, 2020.

[2] The ATLAS Experiment at the CERN Large Hadron Collider. *Journal of Instrumentation*, 3(08):S08003, August 2008.

[3] Standard Model. Available from: `https://home.cern/science/physics/standard-model`.

[4] Andre Sopczak. Searches for Leptoquarks with the ATLAS Detector, 2021. arXiv;2107.10094.

[5] Neuron diagram. Available from: `http://en.wikibooks.org`.

[6] What is a neural network? Available from: `https://www.tibco.com/reference-center/what-is-a-neural-network`.

[7] Decision Tree. Available from: `https://www.smartdraw.com/decision-tree/`.

[8] Sercan O. Arik and Tomas Pfister. Tabnet: Attentive interpretable tabular learning, 2020. arXiv;1908.07442.

[9] Arda Aras. Explaining what learned models predict: In which cases can we trust machine learning models and when is caution required?, March 2020. Available from: `https://www.researchgate.net/publication/350487701_Explaining_what_learned_models_predict_In_which_cases_can_we_trust_machine_learning_models_and_when_is_caution_required`.

[10] Wikimedia: ROC curve diagram. Available from: `https://commons.wikimedia.org/wiki/File:Roc_curve.svg`.

[11] Lukáš Viceník. Machine Learning for the Leptoquark Search Using CERN ATLAS Data, 2022. Available from: `https://cds.cern.ch/record/2812370`.

[12] Leptoquarks and the physics beyond the Standard Model. Available from: https://phys.org/news/2021-12-leptoquarks-physics-standard.html.

[13] The Higgs boson. Available from: https://atlas.cern/Discover/Physics/Higgs.

[14] About CERN. Available from: https://home.cern/about.

[15] The Large Hadron Collider. Available from: https://home.cern/science/accelerators/large-hadron-collider.

[16] The ATLAS Detector. Available from: https://atlas.cern/Discover/Detector.

[17] Concept of luminosity. Available from: https://cds.cern.ch/record/941318/files/p361.pdf.

[18] R. Nave. Quarks. http://hyperphysics.phy-astr.gsu.edu/hbase/Particles/quark.html.

[19] R. Nave. Leptons. http://hyperphysics.phy-astr.gsu.edu/hbase/Particles/lepton.html.

[20] What is Monte Carlo Simulation? Available from: https://www.ibm.com/topics/monte-carlo-simulation.

[21] Ajiboye Abdulraheem, Ruzaini Abdullah Arshah, and Hongwu Qin. Evaluating the effect of dataset size on predictive model using supervised learning technique. *International Journal of Software Engineering Computer Sciences (IJSECS)*, 1:75–84, February 2015.

[22] Mahsana Haleem, Aurelio Juste Rozas, Stergios Kazakos, Masahiro Morinaga, Yasuyuki Okumura, Yoshihiro Shimogama, Tamara Vazquez Schroeder, and Kohei Yorita. Search for Leptoquark pair production decaying to t tau t tau, 2019. Available from: https://cds.cern.ch/record/2695023.

[23] About ROOT Framework. Available from: https://root.cern/about/.

[24] Optuna: A hyperparameter optimization framework. Available from: https://optuna.readthedocs.io/en/stable/.

[25] TRexFitter Docs. Available from: https://trexfitter-docs.web.cern.ch/trexfitter-docs/.

[26] TRexFitter Repository: README. Available from: https://gitlab.cern.ch/TRExStats/TRExFitter/blob/master/README.md.

[27] Template fits: TRexFitter et al. Available from: https://indico.cern.ch/event/822074/contributions/3471458/attachments/1865561/3067487/20190619_TRExFitter_AS.pdf.

[28] Neural Networks and Introduction to Deep Learning. Available from: https://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/st-m-hdstat-rnn-deep-learning.pdf.

[29] Negative log likelihood loss. Available from: https://www.cs.toronto.edu/~rgrosse/courses/csc311_f20/slides/lec06.pdf.

[30] GitHub: Bayesian-Neural-Network-Pytorch. Available from: https://github.com/Harry24k/bayesian-neural-network-pytorch/.

[31] Lior Rokach and Oded Maimon. *Decision Trees, The Data Mining and Knowledge Discovery Handbook*, volume 6, pages 165–192. January 2005.

[32] Lecture 6 - Bagging, Boosting. Available from: https://pytorch.org/docs/stable/generated/torch.nn.NLLLoss.html.

[33] Ensemble Methods: Bagging and Boosting. Available from: https://cse.iitk.ac.in/users/piyush/courses/ml_autumn16/771A_lec21_slides.pdf.

[34] Minghua Chen, Qun-Ying Liu, Shuheng Chen, Yicen Liu, Changhua Zhang, and Ruihua Liu. XGBoost-Based Algorithm Interpretation and Application on Post-Fault Transient Stability Status Prediction of Power System. *IEEE Access*, PP:1–1, 01 2019.

[35] XGBoost Parameters. Available from: https://xgboost.readthedocs.io/en/stable/parameter.html.

[36] Reference for built-in TabNet algorithm. Available from: https://cloud.google.com/ai-platform/training/docs/algorithms/reference/tab-net.

[37] Magician's Corner: 9. Performance Metrics for Machine Learning Models. Available from: https://pubs.rsna.org/doi/pdf/10.1148/ryai.2021200126.

[38] Marina Sokolova, Nathalie Japkowicz, and Stan Szpakowicz. Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation, 01 2006. Available from: https://www.researchgate.net/publication/225215404_Beyond_Accuracy_F-Score_and_ROC_A_Family_of_Discriminant_Measures_for_Performance_Evaluation.

[39] Practical Statistics – part I, Basics Concepts. Available from: https://indico.cern.ch/event/287744/contributions/1641250/attachments/535751/738667/Verkerke_Statistics_1.pdf.

Appendices

Appendix A

Pre-selection

```
(custTrigMatch_LooseID_FCLooseIso_DLT > 0) &&
(dilep_type > 0 && (lep_ID_0*lep_ID_1)>0) &&
((lep_Pt_0>=10e3&&lep_Pt_1>=10e3)&&(fabs(lep_Eta_0)<=2.5&&
fabs(lep_Eta_1)<=2.5)&&
(XXX_TIGHT_PLIV_MUON_0||(XXX_TIGHT_PLIV_ELEC_0 &&
((!(!(lep_Mtrktrk_atConvV_CO_0<0.1&&lep_Mtrktrk_atConvV_CO_0>=0&&
lep_RadiusCO_0>20)&&
(lep_Mtrktrk_atPV_CO_0<0.1 &&lep_Mtrktrk_atPV_CO_0>=0)))&&
!(lep_Mtrktrk_atConvV_CO_0<0.1&&lep_Mtrktrk_atConvV_CO_0>=0&&
lep_RadiusCO_0>20))))&&
(XXX_TIGHT_PLIV_MUON_1 || (XXX_TIGHT_PLIV_ELEC_1&&
((!(!(lep_Mtrktrk_atConvV_CO_1<0.1&&lep_Mtrktrk_atConvV_CO_1>=0&&
lep_RadiusCO_1>20)&&
(lep_Mtrktrk_atPV_CO_1<0.1&&lep_Mtrktrk_atPV_CO_1>=0)))&&
!(lep_Mtrktrk_atConvV_CO_1<0.1&&lep_Mtrktrk_atConvV_CO_1>=0&&
lep_RadiusCO_1>20))))) &&
nTaus_OR==1 &&
nJets_OR_DL1r_85>=1 &&
nJets_OR>=4 &&
((dilep_type==2) || abs(Mll01-91.2e3)>10e3)
```

Appendix B

Python Libraries

acron 0.17.2
alembic 1.7.7
asn1crypto 0.24.0
astroid 1.4.9
atlasplots 0.1.9
attrs 22.2.0
auth-get-sso-cookie 2.2.1
autopage 0.5.1
awkward 1.9.0
awkward0 0.15.5
beautifulsoup4 4.4.1
cachetools 4.2.4
captum 0.6.0
catboost 1.1.1
certmgr-client 1.17.1
cffi 1.9.1
chardet 3.0.4
cliff 3.10.1
cmaes 0.9.0
cmd2 2.4.2
colorlog 6.7.0
compress-pickle 2.1.0
conda 4.6.14
cryptography 2.3
cx-Oracle 7.1.0
cycler 0.11.0
dataclasses 0.8
decorator 4.0.11
DistRDF 6.24.8
distro 1.5.0
docopt 0.6.2
easydict 1.10
elasticsearch6 6.4.2
fts3 3.12.0
gfal2_util 1.8.0
graphviz 0.19.1
greenlet 2.0.1
gssapi 1.3.0
htcondor 9.0.17
html5lib 0.999
idna 2.10
imbalanced-learn 0.8.1
imblearn 0.0
importlib-metadata 4.8.3
importlib-resources 5.4.0
isort 4.2.5
joblib 1.1.1
kaleido 0.2.1
kiwisolver 1.3.1
landbtools 22.9.3
lazy-object-proxy 1.2.2
lightgbm 3.3.3
lxml 4.2.5
M2Crypto 0.35.2
Mako 1.1.6
managesieve 0.6
MarkupSafe 2.0.1
matplotlib 3.3.4
mccabe 0.6.1
megabus 2.1.0
numpy 1.19.5
olefile 0.46
opensearch-py 1.0.0
optuna 2.10.1
packaging 21.3
pandas 1.1.5
pbr 5.11.0
Pillow 6.2.2
pip 21.3.1
plotly 5.11.0
ply 3.9
prettytable 2.5.0
pyasn1 0.4.7
pyasn1-modules 0.2.7
pycosat 0.6.3
pycparser 2.14
pycrypto 2.6.1
pycurl 7.43.0
pylint 1.6.5
PyMySQL 0.9.3
pyOpenSSL 17.3.0
pyparsing 3.0.9
pyperclip 1.8.2
PySocks 1.6.8
python-dateutil 2.8.2
python-ldap 3.1.0
python3-iteslibs 0.7.6
pytorch-tabnet 4.0
pytz 2017.2
PyYAML 3.13
requests 2.14.2
requests-gssapi 1.2.2
ROOT 6.24.8
ruamel.yaml 0.13.14
scikit-learn 0.24.2
scipy 1.5.4
seaborn 0.11.2

setuptools 39.2.0
six 1.14.0
SQLAlchemy 1.4.45
stevedore 3.5.2
suds-jurko 0.6
teigi 4.30.1
tenacity 8.1.0
threadpoolctl 3.1.0
torch 1.10.1
torchbnn 1.2
torchinfo 1.5.4
torchvision 0.11.2
tqdm 4.64.1
typing_extensions 4.1.1
uproot 4.3.7
uproot3 3.14.4
uproot3-methods 0.10.1
urllib-gssapi 1.0.1
urllib3 1.25.6
wcwidth 0.2.5
wget 3.2
wheel 0.31.1
wrapt 1.10.4
xgboost 1.5.2
xlrd 1.0.0
zipp 3.6.0

Appendix C

Confusion Matrices

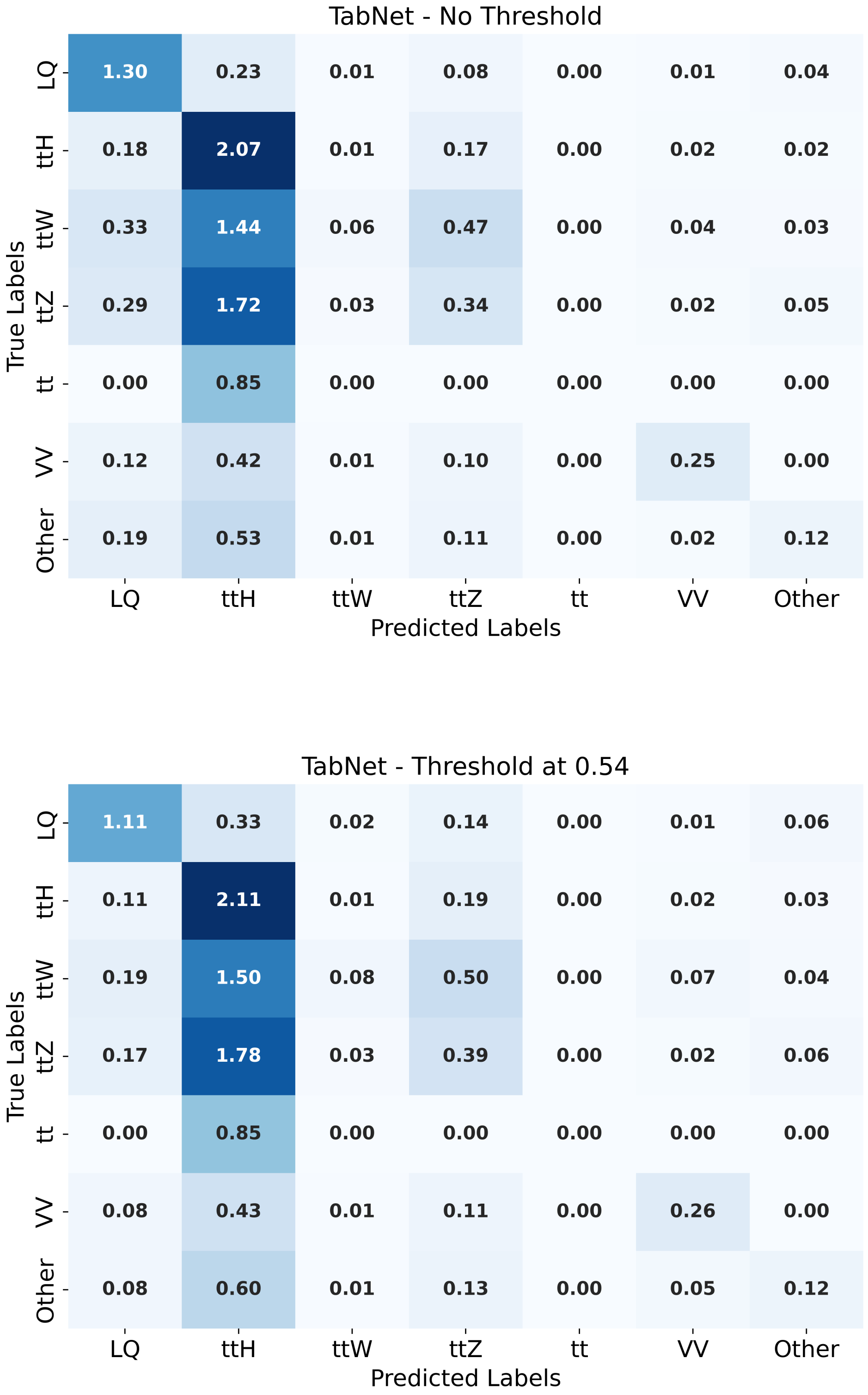


Figure C.1: Confusion matrix for the TabNet model trained on preselected data without using a threshold (upper), and with (lower).

Appendix D

Feature Importance

Feature Name	Mean	Std
taus_pt_0	0.161654	0.020311
sumPsbtag	0.119569	0.030823
MtLepMet	0.092426	0.022242
HT	0.073864	0.022246
DeltaR_min_lep_jet_fwd	0.05193	0.015864
lep_Pt_1	0.042502	0.017427
met_met	0.042161	0.024771
jets_pt_vector_2	0.036962	0.014927
nJets_OR	0.033271	0.024776
MLepMet	0.031118	0.025111
DRll01	0.029379	0.026402
jets_pt_vector_3	0.020594	0.016582
max_eta	0.016976	0.013625
jets_e_vector_0	0.01683	0.012741
lep_E_1	0.016074	0.017901
lep_Pt_0	0.014616	0.012823
taus_RNNJetScoreSigTrans_0	0.014374	0.01455
jets_pt_vector_0	0.014309	0.008256
lep_isolationLoose_VarRad_0	0.013146	0.013088
lep_E_0	0.012272	0.015446
Ptll01	0.011571	0.01204
taus_RNNJetScore_0	0.010704	0.010126
minDeltaR_LJ_0	0.008032	0.010398
HT_lep	0.007982	0.010928
mjjMax_frwdJet	0.006978	0.011639
total_charge	0.006757	0.010764
Mll01	0.006559	0.010181
taus_JetRNNSigTight_0	0.005099	0.007684
lep_ID_0	0.004892	0.00917
taus_DL1r_0	0.004204	0.004378
DeltaR_min_lep_jet	0.004002	0.005226
lep_isolationLoose_VarRad_1	0.003816	0.003669
jets_e_vector_1	0.003306	0.006989
lep_sigd0PV_1	0.003104	0.004336
lep_ID_1	0.003089	0.002967
dEta_maxMjj_frwdjet	0.002981	0.003731
taus_numTrack_0	0.002893	0.00531
taus_eta_0	0.002863	0.003325
taus_fromPV_0	0.002856	0.003214
lep_Z0SinTheta_0	0.00244	0.004501
met_phi	0.002392	0.004979
taus_charge_0	0.00234	0.004229
lep_Eta_0	0.002311	0.004568
jets_eta_vector_0	0.002165	0.003625
lep_nTrackParticles_0	0.002078	0.002784
lep_sigd0PV_0	0.002019	0.002626
lep_Eta_1	0.002003	0.002221
lep_nTrackParticles_1	0.00195	0.002602
jets_eta_vector_2	0.001905	0.002912
minDeltaR_LJ_2	0.00189	0.002857
taus_width_0	0.001874	0.002016
minDeltaR_LJ_1	0.001684	0.00242
jets_e_vector_3	0.001678	0.003066
taus_phi_0	0.001675	0.002744
lep_Z0SinTheta_1	0.001456	0.002365
jets_phi_vector_1	0.001428	0.001674
jets_phi_vector_3	0.001178	0.001556
lep_EtaBE2_1	0.001116	0.001694
dilep_type	0.001062	0.001305
HT_fwdJets	0.000922	0.001012

Table D.1: Feature importance of top 60 features of the selected TabNet model trained on 71 selected features, using all LQ masses, using information from 20 trials.

Feature Name	Mean	Std
taus_pt_0	0.239648	0.01565
MtLepMet	0.135466	0.018085
sumPsbtag	0.053192	0.00332
HT	0.026984	0.003868
Mll01	0.017717	0.001202
taus_charge_0	0.017004	0.00386
taus_numTrack_0	0.015165	0.002016
nJets_OR	0.0149	0.001486
HT_inclFwdJets	0.013736	0.002853
DeltaR_min_lep_jet_fwd	0.013664	0.001772
met_met	0.012342	0.001261
lep_ID_0	0.012295	0.001733
minDeltaR_LJ_0	0.011836	0.00144
lep_Pt_1	0.011285	0.001103
DeltaR_min_lep_jet	0.011269	0.002011
MLepMet	0.010992	0.001156
lep_E_1	0.010441	0.001666
lep_Eta_1	0.010243	0.000981
taus_RNNJetScore_0	0.010183	0.0007
jets_pt_VECTORIZED_3	0.010131	0.000843
minDeltaR_LJ_1	0.009992	0.000816
taus_eta_0	0.009368	0.000384
max_eta	0.009249	0.001032
lep_E_0	0.009189	0.001429
taus_JetRNNSigTight_0	0.009124	0.00856
HT_lep	0.009124	0.001242
jets_pt_VECTORIZED_2	0.00907	0.000814
DRll01	0.008862	0.000429
HT_fwdJets	0.008844	0.000657
Ptll01	0.008728	0.000665
taus_width_0	0.008427	0.000456
taus_DL1r_0	0.00839	0.000665
lep_ID_1	0.00806	0.001899
lep_Eta_0	0.007906	0.000547
lep_sigd0PV_0	0.007882	0.000487
jets_pt_VECTORIZED_1	0.007742	0.000662
lep_sigd0PV_1	0.00772	0.000405
mjjMax_frwdJet	0.007708	0.000457
taus_fromPV_0	0.007577	0.002314
taus_RNNJetScoreSigTrans_0	0.00748	0.000768
lep_Pt_0	0.007471	0.000596
dEta_maxMjj_frwdjet	0.007242	0.000402
jets_e_VECTORIZED_0	0.007112	0.000526
lep_Z0SinTheta_1	0.006724	0.000495
minDeltaR_LJ_2	0.006605	0.000463
taus_phi_0	0.006369	0.00046
jets_eta_VECTORIZED_3	0.006288	0.000437
jets_e_VECTORIZED_1	0.006191	0.000447
lep_Phi_0	0.006184	0.000461
lep_EtaBE2_0	0.006164	0.000542
lep_Z0SinTheta_0	0.006146	0.000327
jets_pt_VECTORIZED_0	0.006097	0.000353
jets_eta_VECTORIZED_2	0.006061	0.000452
eta_frwdjet	0.00604	0.000576
lep_Phi_1	0.005978	0.000479
met_phi	0.005857	0.000453
jets_e_VECTORIZED_3	0.005847	0.00039
jets_e_VECTORIZED_2	0.005666	0.000479
dilep_type	0.005457	0.001507
jets_eta_VECTORIZED_1	0.005434	0.000594

Table D.2: Feature importance of top 60 features of the selected XGBoost model trained on 71 selected features, using all LQ masses, using information from 20 trials.

Appendix E

Feature Comparison of Different Masses

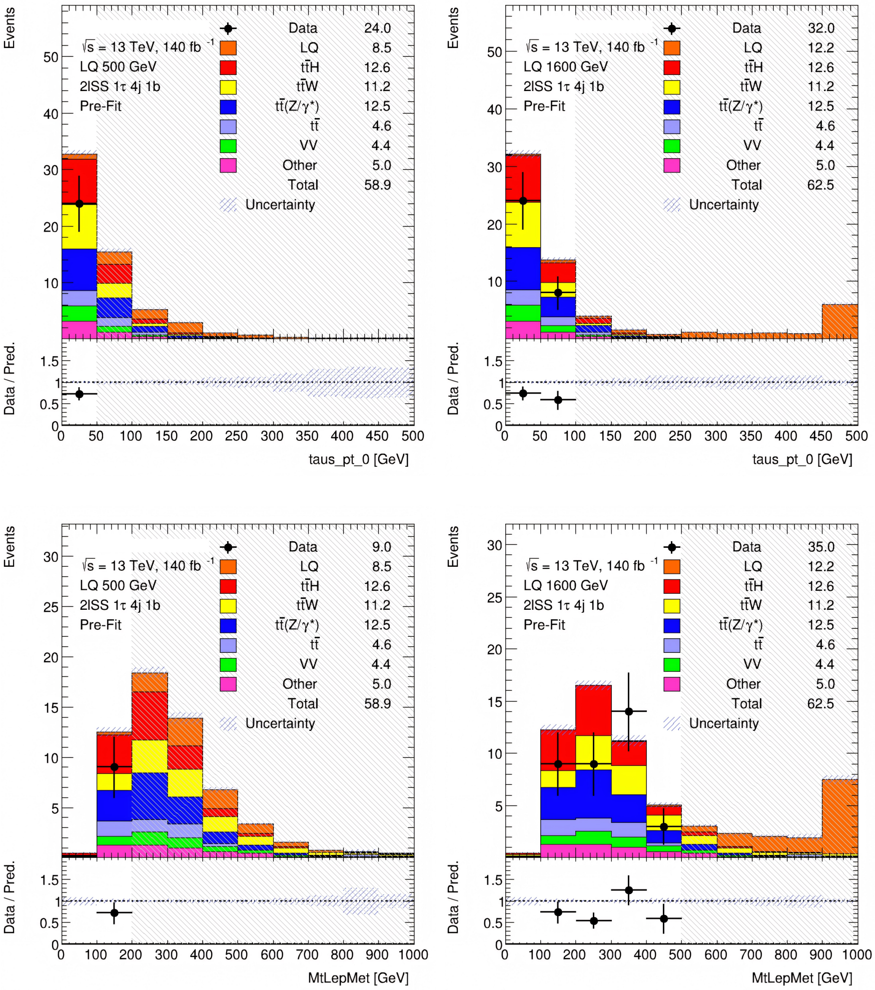
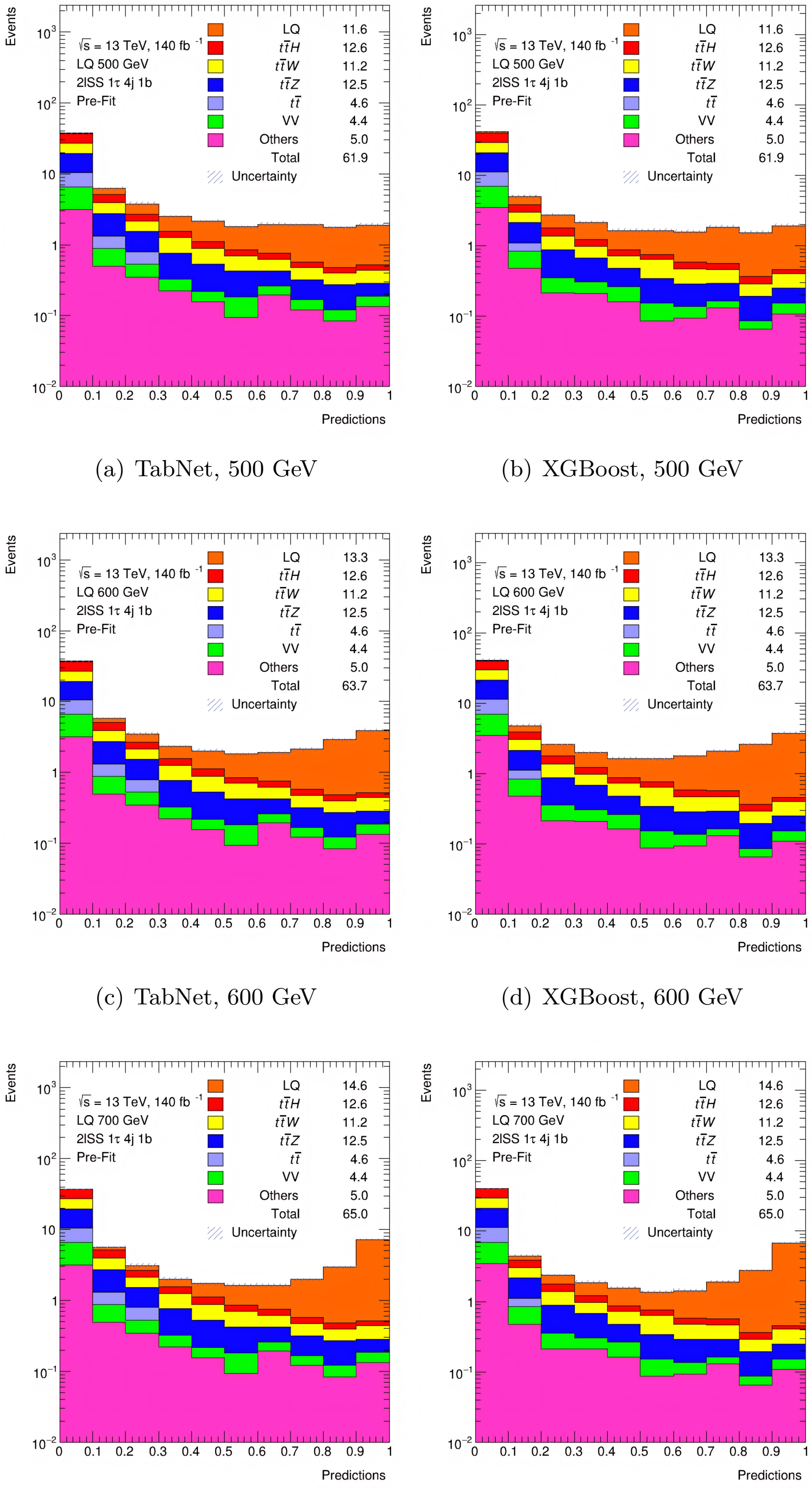


Figure E.1: Histograms with distributions of two features with high importance for 500 GeV (left) and 1600 GeV (right), with LQ $\sigma = 0.01\,\text{pb}$.

Appendix F

Network Output



(a) TabNet, 500 GeV

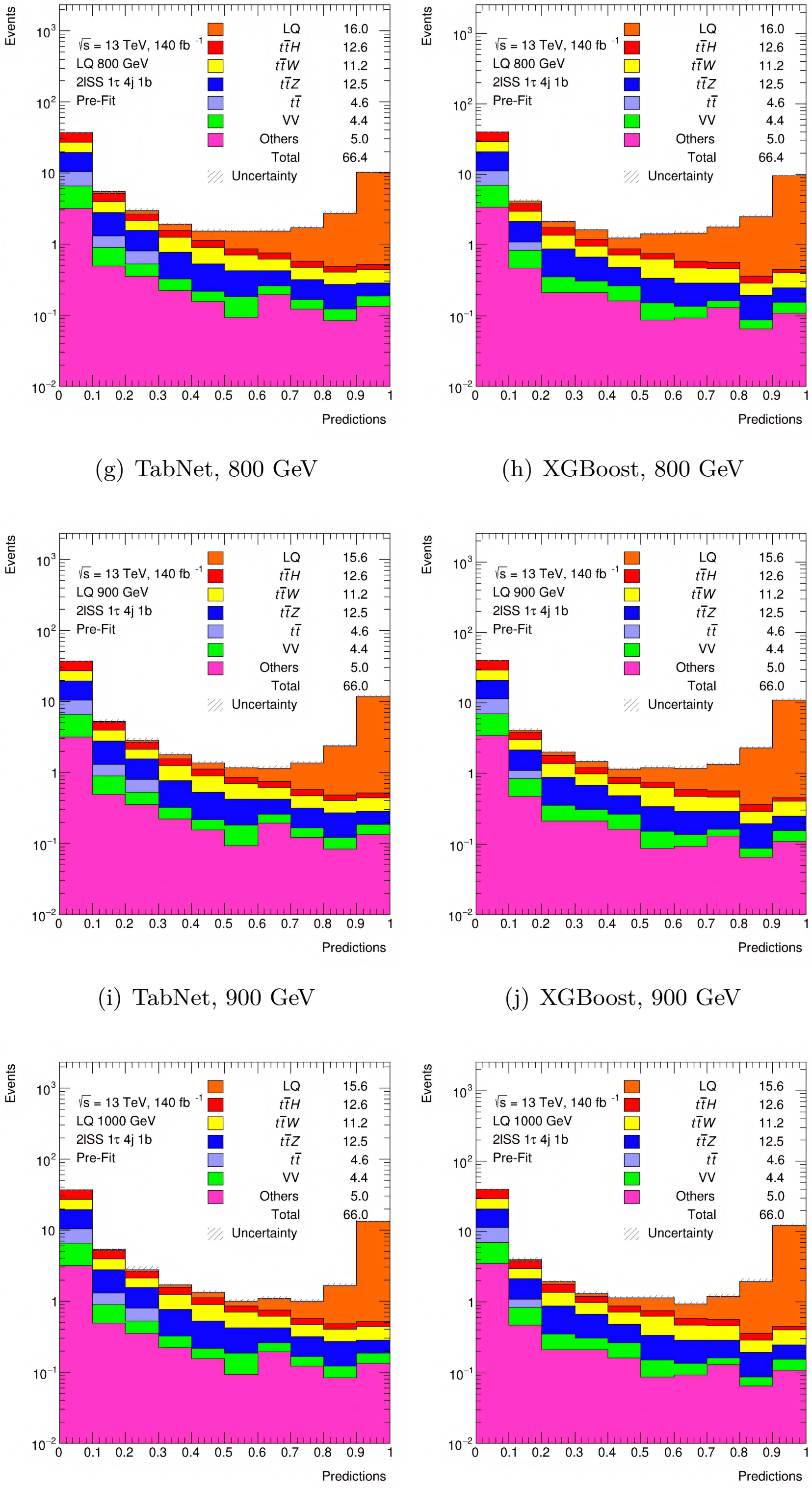
(b) XGBoost, 500 GeV

(c) TabNet, 600 GeV

(d) XGBoost, 600 GeV

(e) TabNet, 700 GeV

(f) XGBoost, 700 GeV



(g) TabNet, 800 GeV

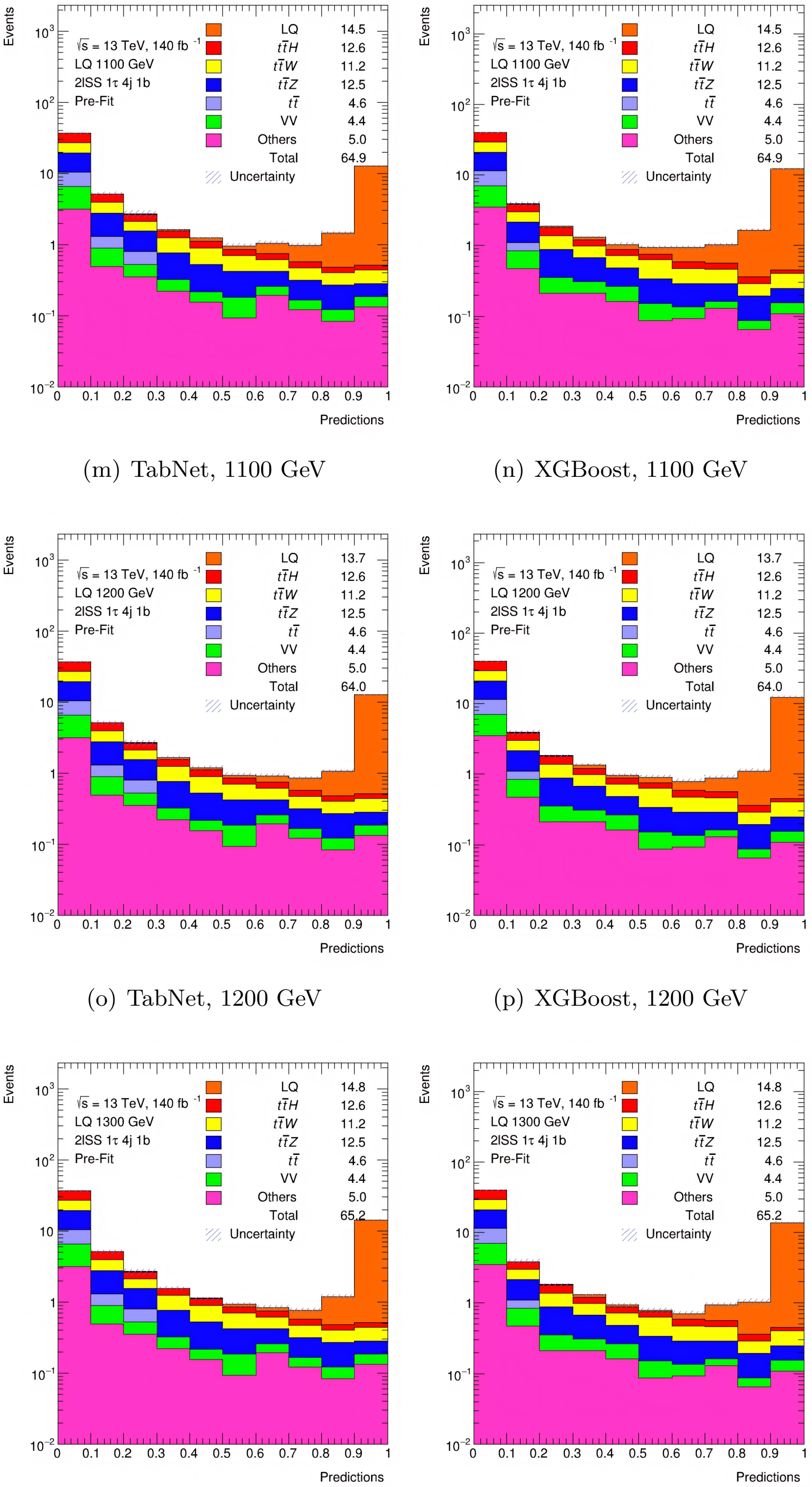
(h) XGBoost, 800 GeV

(i) TabNet, 900 GeV

(j) XGBoost, 900 GeV

(k) TabNet, 1000 GeV

(l) XGBoost, 1000 GeV



(m) TabNet, 1100 GeV

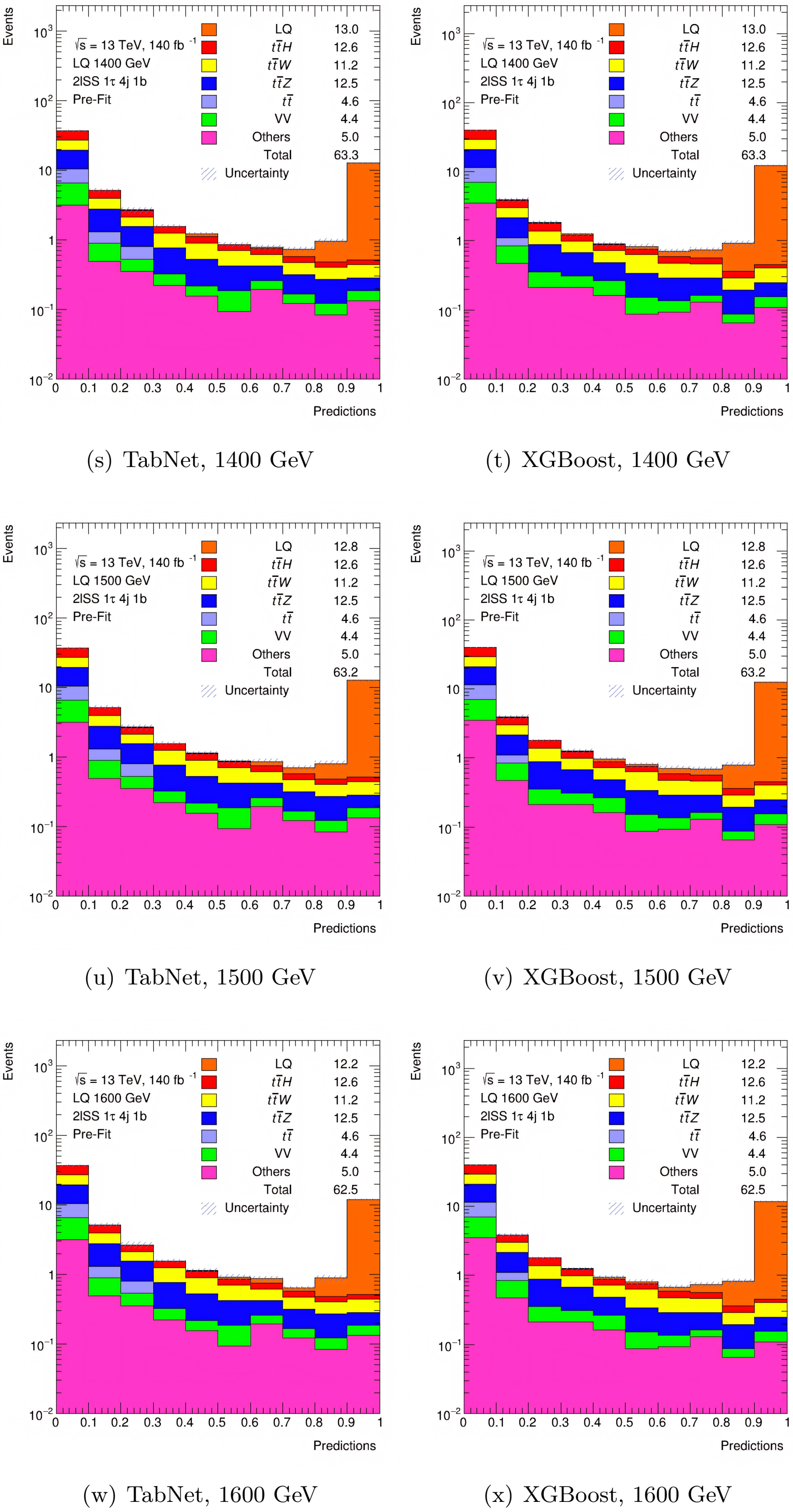
(n) XGBoost, 1100 GeV

(o) TabNet, 1200 GeV

(p) XGBoost, 1200 GeV

(q) TabNet, 1300 GeV

(r) XGBoost, 1300 GeV



(s) TabNet, 1400 GeV

(t) XGBoost, 1400 GeV

(u) TabNet, 1500 GeV

(v) XGBoost, 1500 GeV

(w) TabNet, 1600 GeV

(x) XGBoost, 1600 GeV

Figure F.1: Network output corresponding to the signal class for all masses of both TabNet and XGBoost models, with LQ $\sigma = 0.01\,\text{pb}$.

Appendix G

Binary Classifier Results

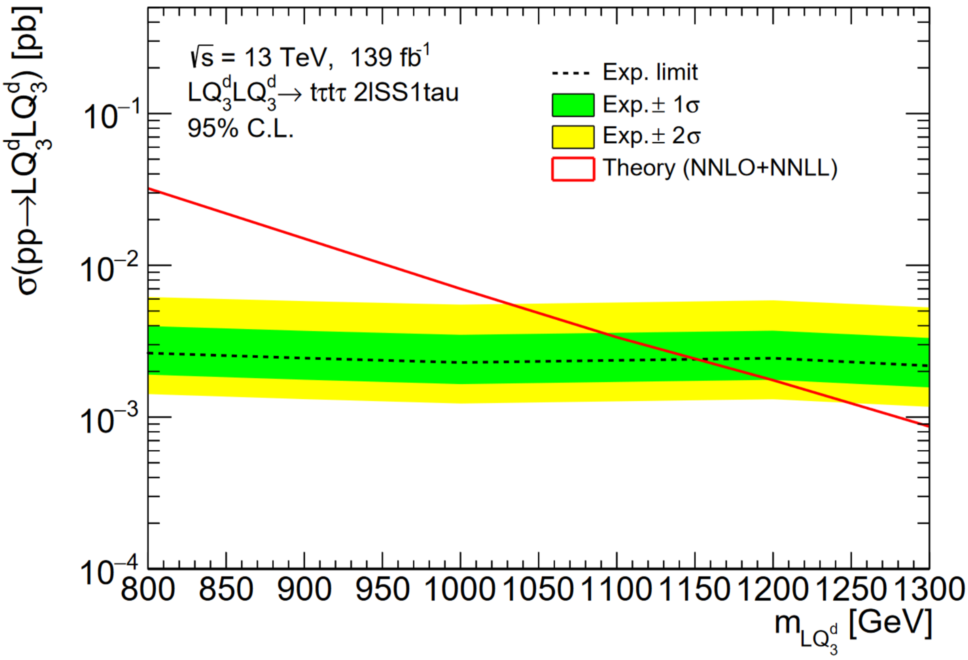


Figure G.1: Expected upper limit plots with logarithmic scale, obtained using the XGBoost binary classifier trained on data without preselection.

Appendix H

TRExFitter Config for Cross-section Estimate

```
Job: "XXX_JOB_NAME"
  CmeLabel: "13 TeV"
  ReadFrom: NTUP
  POI: "norm_LQ"
  NtuplePaths: XXX_NTUPLE_PATHS
  NtupleFiles: XXX_NTUPLE_FILES
  NtupleName: nominal
  OutputDir: output_limit
  Label: "2lSS1Tau inclusive"
  LumiLabel: "140 fb^{-1}"
  Lumi: XXX_LUMI % pb^-1
  PlotOptions: YIELDS,CHI2
  GetChi2: TRUE
  DebugLevel: 10
  HistoChecks: NOCRASH
  ImageFormat: "png","eps"
  ReplacementFile: replacement.txt
  SplitHistoFiles: TRUE
  BlindingThreshold: 0.20

NormFactor: "norm_LQ"
  Title: "norm_LQ"
  Nominal: 1
  Samples: LQ*

Limit: "limit"
  LimitType: ASYMPTOTIC

Significance: "significance"
```

```
Region: "SR"
  Type: SIGNAL
  DataType: ASIMOV
  Label: "xs=1 pb"
  Variable: "y_proba",25,0,1
  VariableTitle: "Predictions"
  LogScale: TRUE

Sample: "LQ_MASS"
  Type: SIGNAL
  Title: "LQ MASS"
  TexTitle: "$LQ MASS"
  FillColor: 807
  LineColor: 1
  Group: "LQ"
  Selection: "XXX_LQ_SELECTION"
  MCweight: "XXX_MCWEIGHT" * "XXX_LQ_MASS_SCALE"

Sample: "ttH"
  Type: BACKGROUND
  Title: "#it{t#bar{t}H}"
  TexTitle: "$t\bar{t}H$"
  FillColor: 632
  LineColor: 1
  Selection: "XXX_ttH_SELECTION"
  MCweight: "XXX_MCWEIGHT" * "XXX_ttH_SCALE"
```

Appendix I

Expected Upper Limit

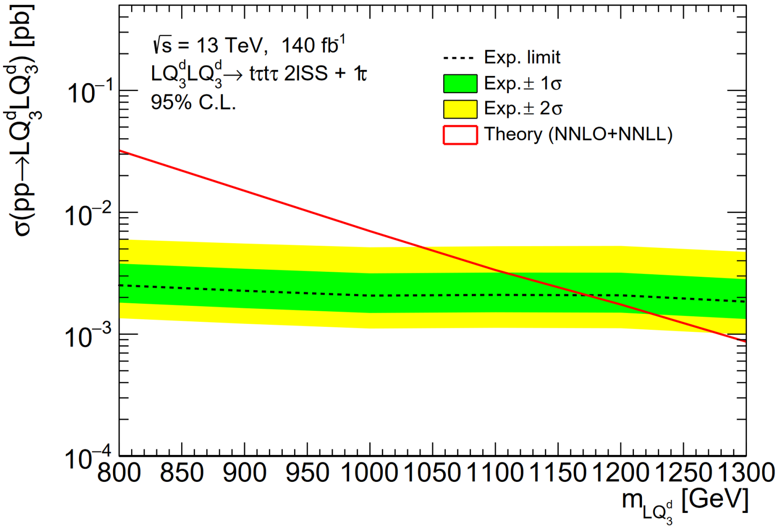


Figure I.1: Expected upper limit plots with logarithmic scale, obtained using the XGBoost model trained on data without preselection.

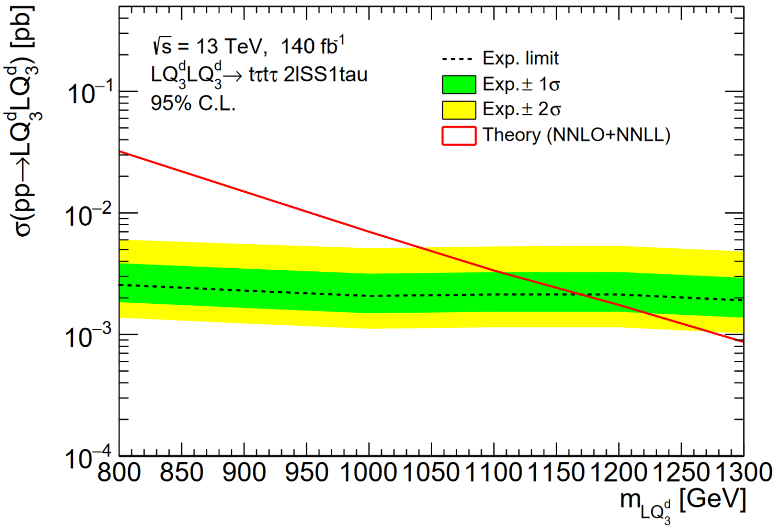
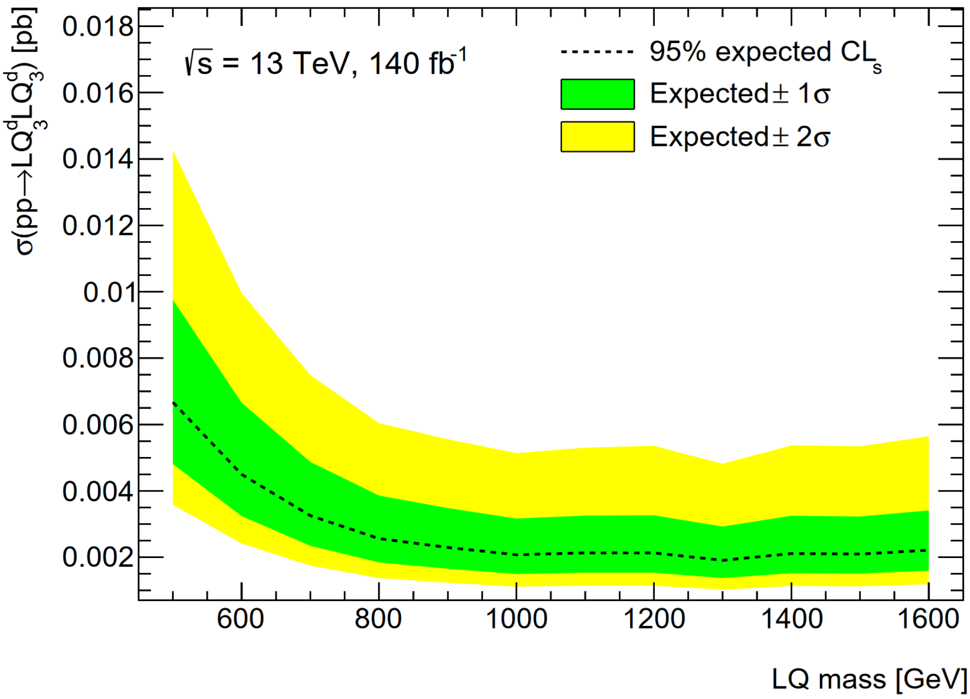


Figure I.2: Expected upper limit plots with logarithmic scale, obtained using the TabNet model trained on data without preselection.



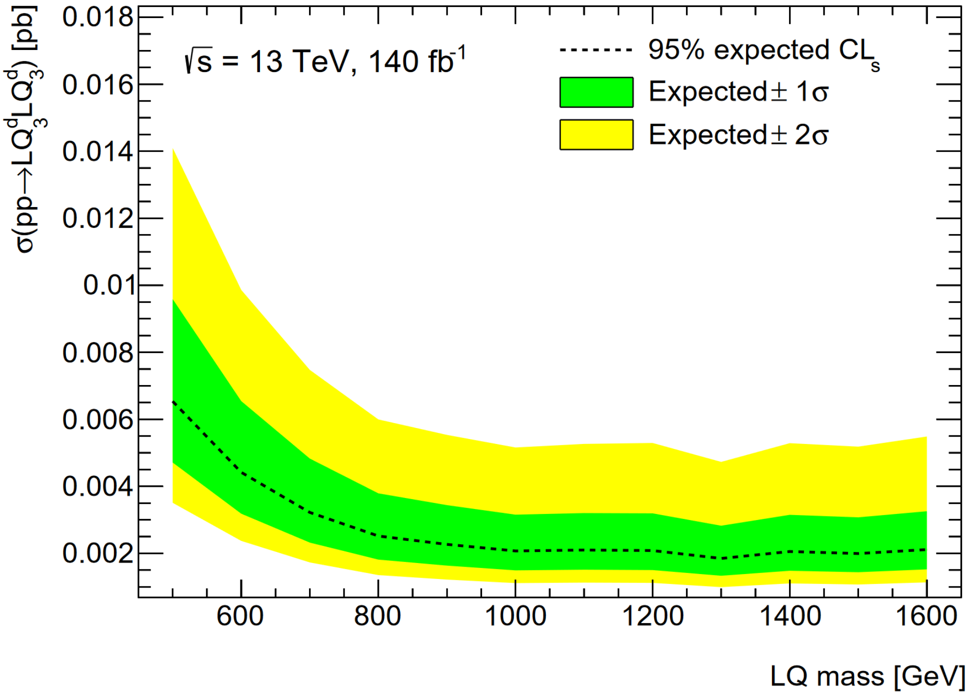


Figure I.3: Expected upper limit plots with linear scale, obtained using the TabNet model (upper) and XGBoost model (lower) trained on data without preselection.