Search for heavy neutrinos in a 3-lepton final-state

Applications using supervised machine learning

by

William Hirst

THESIS

for the degree of

MASTER OF SCIENCE

Faculty of Mathematics and Natural Sciences
University of Oslo

Autumn 2022

Search for heavy neutrinos in a 3-lepton final-state

Applications using supervised machine learning

William Hirst

Search for heavy neutrinos in a 3-lepton final-state

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

Abstract

This will be the abstract.

Acknowledgments

Halla

Contents

Introduction **1**

1 The Standard model of elementary particles and beyond. **3**
1.1 The bulding blocks 3
1.1.1 The leptons 4
1.1.2 The quarks 4
1.2 The Forces 5
1.2.1 Quantum electro dynamics 5
1.2.2 The Weak Force 5
1.2.3 The Strong Force 5
1.3 Beyond the Standard Model 5
1.3.1 Why look beyond? 5
1.3.2 Neutrino-Mass problem 5
1.4 The Background Channels 6
1.5 The Signal 7

2 Introduction to machine learning and statistical analysis **9**
2.1 Machine Learning 9
2.1.1 Phenomenology 9
2.1.2 Neural Networks 9
2.1.3 Gradient Boosting and decision trees 13
2.2 Statistical and multi-variabel analysis 14

3 Implementation **15**
3.1 Tools and Data 15
3.1.1 Monte Carlo Data 15
3.1.2 ROOT, RDataframe and Pandas 15
3.1.3 Computing features in ROOT: Example 16
3.2 Features 17
3.2.1 Cuts and triggers 17
3.2.2 Lepton variables selection 18
3.2.3 Jet variables selection 19
3.2.4 Validation 19
3.2.5 Negative Weights 21

Appendices **29**

Appendix A **31**

Introduction

The Standard Model (SM) is perhaps one of the most successful scientific theories ever created. It accurately explains the interactions of leptons and quarks as well as the force carrying particles which mediate said interactions. In 2012 the SM achieved one of its crowning achievements when we discovered the Higgs boson. Much of the accolade was rightfully given to the theoretical work on the SM, but another aspect of the discovery was equally important. Data analysis was and is a crucial part of any new discovery in physics. One of the most important and exiting tools is Machine Learning (ML).

Outline of the Thesis

Chapter 1

The Standard model of elementary particles and beyond.

The SM is the most successful scientific theory ever created. It accurately explains the interactions of leptons and quarks as well as the force carrying particles which mediate said interactions. The model is a result of over a century of work demanding the contributions of grat minds like Paul Dirac, Erwin Schrodinger and Richard Feynman. In 2012 the SM achieved one of its crowning achievements when we discovered the Higgs boson.

1.1 The bulding blocks

As early as ancienct greece, humans pondered the nature of the most elementary building blocks of the universe. They imagined a rope of a given length, with a pair of scissors of adjustable size. Then one could ask, how many times can you cut the rope in half? If the answer is less than infinite, what are you left with?
In 1897, Joseph John Thomson (*1856-1940*) discovered the first elementary particle using the Cathode Ray Tube. This particle we later named the electron. Prior to the time of discovery, we believed atoms to be the smallest building blocks. After the discovery of the electron, the discovery of the proton and neutron quickly followed. It was not until more than 50 years after the discovery of the proton (by Ernest Rutherford) that we discovered that also protons and neutrons could be further dissected to smaller particles. We call these particles quarks. The "final-piece"[1] of the puzzle came in 1956 when we discovered the, at that time thought of as massless neutrino. Together with the electron, the neutrino is defined as a lepton. Together with the quarks and leptons are called fermions.
Upon the evolution of the quantum mechanics and a physics as a whole, we started to divert our focus from the what and over to the how. How can we explain all the complex interactions that emerge between these relatively simple particles? Through the creation of SM and countless experiment, we discovered that forces are nothing but interactions between particles and fields. The SM describes all forces as a field which are mediated through a particle, we call bosons.
The four forces responsible for all the forces in the universe are electromagnetism (Quantum Electro Dynamics (QED)), the weak-force, the strong-force(Quantum Chromo Dynamics (QCD)) and gravity. The boson most familiar to most is the photon. The photon is responsible for the mediation of QED and is responsible for all electromagnetic effect, such as the ones allowing us to sea objects using our eyes. Similarly, the W and Z bosons are responsible for the weak-force which allows for radioactive decay. And the gluon is responsible for QCD which holds protons and neutrinos together. Gravity is the only force not described in the SM, but would (if one day included) have its own force carrying particle, graviton.
The final building block in the universe introduced and described by SM is the Higgs boson. The Higgs boson was proposed by Peter Higgs in 1964 and discovered at CERN in 2012. The Higgs boson, also called the God particle is responsible for giving particles mass in a process called spontaneous symmetry breaking (more on this in later sections). Together the fermions and the bosons make up all the particles in the SM as it now stands.

[1] Given the nature of this thesis, the existence of further pieces is implied.

Generation	*Flavour*	Mass [MeV]	Charge [Elementary charge]
$1st$	e	0.511	-1
$1st$	ν_e	< 0.001	0
$2nd$	μ	105.66	-1
$2nd$	ν_μ	< 0.17	0
$3rd$	τ	1776.8	-1
$3rd$	ν_τ	< 18.2	0

Table 1.1: A list of all leptons along with their generation, flavor, mass and charge.

Generation	*Flavour*	Mass [MeV]	Charge [Elementary charge]
$1st$	u	2.2	$-2/3$
$1st$	d	4.7	$+1/3$
$2nd$	c	1280	$-2/3$
$2nd$	s	96	$+1/3$
$3rd$	t	173100	$-2/3$
$3rd$	d	4180	$+1/3$

Table 1.2: A list of all quarks along with their generation, flavor, mass and charge.

1.1.1 The leptons

The leptons are all elementary particle with half-integer spin, $\pm 1/2$. A lepton can either be charged or neutral. For reasons that are yet to be known, the leptons come in 3 generations. Each generation containing a pair of charged and neutral lepton. The first generation contains the electron, e^- and the electron-neutrino, ν_e. The second contains the muon, μ and the muon-neutrino, ν_μ. And the third generation contain the tau, τ^- and ν_τ. The generations are numbered by the mass of the charged lepton, where the first generation is the lightest. As is often the case in particle physics, the heavier a particle, the rarer. This is due to the heavier particles (higher generations) quickly decaying into lighter particles (lower generation), in a process we call particle decay. This explains why particle physicists neglect the τ when speaking about leptons, given that this is by far the heaviest and also the rarest.
The charged leptons are all massive particles ranging from a fraction of 1eV to more than a thousand times that. The neutrinos were up until the turn of the millennia presumed to be massless. This was not only backed by experiments but also by the SM which seldom seemed to be wrong. In 1998 it was discovered that neutrinos in fact do have mass although being extremely light. Given the size of the masses we are yet to accurately measure the mass of the neutrinos, but we have found them all to be less than 20 MeV. The fact that the neutrinos in fact do have mass is a problem which will be discussed further in later section. In table 1.1, a summary of all leptons are found, along with the respective mass and charge.

1.1.2 The quarks

> 'Three quarks for Muster Mark!
> Sure he hasn't got much of a bark.
> And sure any he has it's all beside the mark.' [1]

The poem above was written by James Joyce in 1939, and was the motivation for Gell-Mann coining the inner particles of hadrons, quarks. Quarks were introduced to explain some strong-force properties of hadrons. We categorize quarks as either up- or down quarks. All up-quarks have a negative charge equal to 2/3 that of the electron and all down quarks have a positive charge equal to 1/3 that of the electron. Similarly to leptons, all quarks have a spin equal to 1/2 and are divided in 3 generations. Each generation of quarks are made of a pair of one up- and one down-quark. The first generation contains the up, u and the down, d quark, the second the charm, c and the strange, s quark and third the top, t and the bottom, b quark. Also similarly to leptons, the higher the generation and mass the rarer the quarks. Similarly to how difference in spin allows leptons to stay in an otherwise similar quantum state, the quarks have color. The colors of quarks are what connects them to the strong-force. The strong force is what allows quarks to change color and also explains a phenomenon known as

color confinement. Briefly explained, color confinement results in quarks never existing in isolation but always in pairs, mesons or in threes, baryons (like protons and neutrons). Given color confinement, quarks are never directly observed in experiments, instead we detect the signature of quarks forming hadrons in a process called hadronisation. We call these signature jets.

1.2 The Forces

1.2.1 Quantum electro dynamics

1.2.2 The Weak Force

1.2.3 The Strong Force

1.3 Beyond the Standard Model

1.3.1 Why look beyond?

> 'There is nothing new to be discovered in physics now.
> All that remains is more and more precise measurement.' [2]

The quote above is rumored to have been spoken by William Thompson(*1824–1907*), better known as Lord Kelvin when addressing the British Association for the Advancement of Science in 1900. The statement followed a long period of advancements in the field of physics by the likes of James Clerk Maxwell (*1831–1879*) and Michael Faraday (*1791–1867*). It would take less than half a decade before he would understand the magnitude of his miscalculation, when Einstein and Planck began the development of Quantum Mechanics. Just as Kelvin was wrong back then, would he be wrong today. For although SM explains a large range of phenomena, there are yet many mysteries to explain in the universe and even problems rooted in SM. In this section I will explain some of the problems we hope to tackle in the future.

As mentioned in previous section, SM is yet to explain *gravity*. The hope has been to integrate gravity into SM through the discovery of a gravity-carrying particle, the gravtion. So far, no-such particle is found. *Dark matter* and *dark energy* are also not described by SM, even though the two makeup more than 90% of the mass in the observable universe. Inflation is today the leading explanation to what happened in the early-stages (the first fraction of a second) of the universe. It explains a universe in which all space undergoes a rapid increase in rate of expansion. None of the fields explained by SM are capable of causing any such expansion. Finally, and the one most relevant for this analysis is the neutrino-mass and Charge-Parity (CP)-violation problems, but this will be discussed in the next section.

1.3.2 Neutrino-Mass problem

Neutrinos have a special place in physics. For one, they are the only particles that only interact by the weak force. This means that neutrinos rarely interact at all. It is often used as a (granted for many not incredibly exiting) conversation piece that a colossal amount ($ca.10^{14}$) of neutrinos pass through your body every second. This is harmless to us exactly because of the rarity of neutrinos interaction with anything. For this reason we call neutrinos ghostly.
Another reason neutrinos are special is that they exhibit flavor mixing. In 2015 Arthur McDonald (*1943-*) and Takaaki Kajita (*1959-*) were awarded the Nobel Prize for their contributions in the discovery. In simple terms, flavor mixing is a process where a particle oscillates between different flavors. For neutrinos this means oscillating between ν_e, ν_μ and ν_τ. Flavor mixing is in it of itself not special. Quarks have been observed to exhibit the same behavior. The reason this is interesting in the case of neutrinos, is that it implies that neutrinos are massive. Before this discovery, we believed neutrinos to be massless. But why are massive neutrinos a problem?
All (previously) known massive particles gain mass through interactions with the Higgs boson. For particles to gain mass they need to have a right- and left-handed[2] particle. So far, no right-handed neutrinos have been

[2]The handedness of a particle is defined as a relation between the direction of spin and momentum for a particle. Right means the two are directed in the same direction and left corresponds to opposite direction.

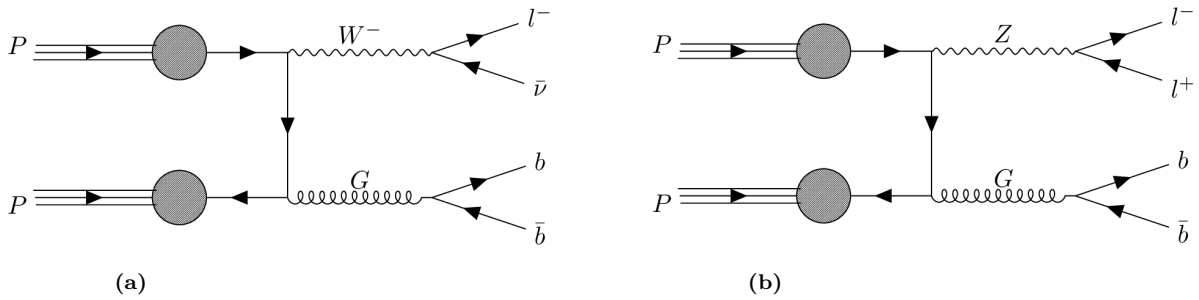


Figure 1.1: The Feynman diagram of both the W+jets 1.1a and the Z+jets 1.1b.

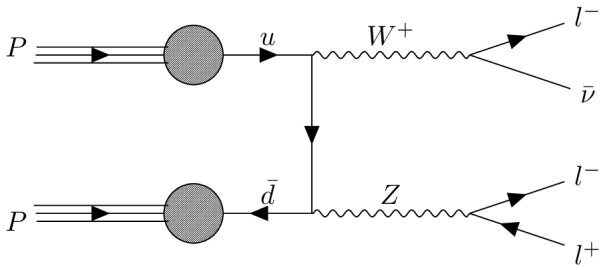


Figure 1.2: The Feynman diagram of the diboson WZ-channel.

observed. Generally there are two schools of thought for why a right-handed neutrino has not been observed. Either, it is very heavy, and we are yet to generate energies large enough to recreate it, or it is indistinguishable to the left-handed neutrino. In the first scenario we would call the right-handed neutrino a *Dirac* fermion. This means that it is no different from any other right-handed lepton. In the second the right-handed neutrino is a *Majorana* fermion. A Majorana fermion is simply a lepton where the particle and the antiparticle are the same. In this thesis I have searched for a right-handed neutrino and considered both Majorana and Dirac neutrinos.

1.4 The Background Channels

The dominant SM backgrounds can be divided into two categories: (i) from leptonic τ decays and (ii) from fake leptons. In the first category, the dominant process is the pair production of WZ with W decaying leptonically and $Z \to \tau\tau$. The trilepton final states with no-OSSF pairs can arise from the subsequent leptonic decay of τ 's. We estimate this background process via Monte Carlo simulations.
The dominant processes of the second category are $\gamma^*/Z+$ jets and $t\bar{t}$, where two leptons come from $\gamma^*/Z \to \tau\tau$ or the prompt decay of t and $\bar{t}$, and a third lepton is faked from jets containing heavy-flavor mesons.

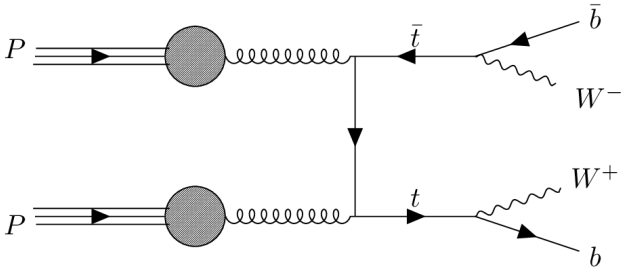


Figure 1.3: The Feynman diagram of the $t\bar{t}$-channel.

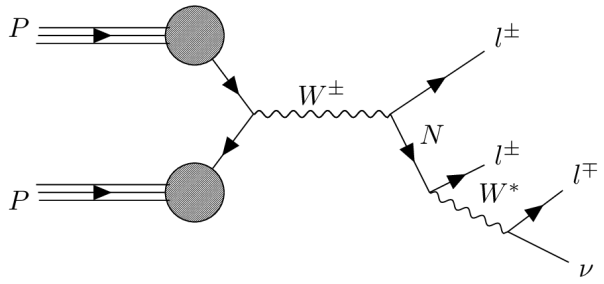


Figure 1.4: The Feynman diagram of the signal-channel.

1.5 The Signal

Chapter 2

Introduction to machine learning and statistical analysis

Machine learning is rapidly becomming an overwhelming presens in many different scientific fields. In areas ranging from cancer research to stock-trading, machine learning is being applied to probelms once thought as impossible to solve. Particle physics, like many other fields is no exeption. Jet flavor classification [3], separating jets from gluons [4] or using ML to create efficent Signal region (SR) are just some examples where ML is a vital tool. The traditional approach for ML in high-energy physics is through the use of supperivsed learning. Deep Neural Networks (DNN)

2.1 Machine Learning

2.1.1 Phenomenology

ML differs from other analysis tools in its ability to learn. Where a purely analytical model is static in both method and performance a ML model aims to be dynamic and self improving. ML utilizes data to leverage towards an ideal model. The extent of utilization defines a ML model as being either supervised, semi-supervised or unsupervised. In the case of supervised ML a set of targets are provided along with the data which allows a ML model to map directly from data to target. In the case of unsupervised ML, no target is provided. Semi-unsupervised is a loose term which finds it self in the middel of the previous two. It often refers to methods where no concrete target is provided, but instead uses the data provided to create a target

2.1.2 Neural Networks

I will now introduce the basic concepts of the Neural Network (NN). Specifically I am going to address a dense feed forward NN, meaning the information is only moving forward trough the network and each node in a given layer is connected to each and every node in the following layer. In the beginning we have an input layer containing one node for each feature of the data. The following layers are the so-called hidden layers which can have a custom chosen number of nodes for each layer. The number of hidden layers and nodes in each respective hidden layers defines the complexity and depth of the NN. A rule of thumb in computer science is that more than 3 hidden layers is defined as a deep-NN. Finally we have the output layers with one node per target category. For a simple True and False type of problem a single output node which takes values between 0 and 1 is sufficient for the prediction. For classification between multiple classes we would require an output node for each class. The connection between the nodes is controlled by the introduction of the weights w and biases b. The weights and biases serve as the unknown parameter. For an already trained network we can collect the predictions by performing the so-called feed forward mechanism which is explained in details in the following section.
However, before diving into the details behind the neural network and the key algorithms for performing training and prediction, we want to put down some notation. We aim to use some of the most common notation, but it is nonetheless easy to get lost. Thus we provide a table (2.1) for which you can always go back to clarify the meaning of variables and indexes.

Table 2.1: Notation

Matrices and vectors		
Notation	Description	Type
X	Design Matrix (input data).	$\mathbb{R}^{N\times\#\text{features}}$
t	Target values.	$\mathbb{R}^{N\times\#\text{categories}}$
y	Model output, the prediction from our network.	$\mathbb{R}^{N\times\#\text{categories}}$
W^l	The weight matrix associated with layer l which handles the connections between layer $l-1$ and l .	$\mathbb{R}^{n_{l-1}\times n_l}$
B^l	The bias vector associated with layer l which handles the biases for all nodes in layer l.	$\mathbb{R}^{n_l\times 1}$
Elements		
w^l_{ij}	The weight connecting node i in layer $l-1$ to node k in layer l.	$\mathbb{R}$
b^l_j	Bias acting on node j in layer l.	$\mathbb{R}$
z^l_j	Node output before activation on node j on layer l.	
a^l_j	Activated node output on node j on layer l.	$\mathbb{R}$
Functions		
C	Cost function	
σ^l	Activation function associated with layer l.	
Quantities		
n_l	The number of nodes in layer l.	
L	Number of layers in total with $L-2$ hidden layers.	
N	Total number of data points.	
All indexing starts from 1: $i, j, k, l = 1, 2, \ldots$		

Feeding Forward

The forward mechanism is the main mechanism behind the usage of the NN. For a single data point the data flow becomes as outlined in the following.

1. The input nodes receive the input data for each feature.
2. Each input node sends the data value into each node of the following hidden layer with a scaling according to the associated weight of every single connection.

3. Each node in the hidden layer sums up the weighted contributions from the input layer and adds a bias value associated to that given node. We denote this raw node output by z.
4. The unactivated value z is immediately sent through an activation function σ associated with the layer to produce the activated value $a = \sigma(z)$.
5. Each node in the hidden layer then forwards the activated value into the next layer using the same procedure. Notice that the number of nodes in the hidden layers no longer corresponds to any given features and one can only speculate on how the complex network creates its own sub-features and interprets the underlying correlations in the data.
6. Finally the stream of forwarded data enters the output layer where the activation values on each node corresponds to the network predictions for each category.

The feed forward step to obtain the activation value on a given layer $l \neq 1$ (not the input layer) is given as

$$z_j^l = \sum_{i=1}^{n_{l-1}} w_{ij}^l a_i^{l-1} + b_j^l, \quad a_j^l = \sigma^l(z_j^l) \tag{2.1}$$

where z_j^l is the input value to node j in layer l, w_{ij}^l is the weight connecting node i in layer $l-1$ to node j in layer l, a_i^{l-1} is the activation value from node i in layer $l-1$ and b_j^l is the bias associated with node j in layer l. We can obtain a matrix-variant of equation 2.1 by defining

$$W^l = \begin{bmatrix} w_{11}^l & w_{21}^l & \dots & w_{n_{l-1}1}^l \\ w_{12}^l & w_{22}^l & \dots & w_{n_{l-1}2}^l \\ \vdots & \vdots & & \vdots \\ w_{1n_l}^l & w_{2n_l}^l & \dots & w_{n_l n_{l-1}}^l \end{bmatrix},$$

$$(W^l)^T = \begin{bmatrix} w_{11}^l & w_{12}^l & \dots & w_{1n_l}^l \\ w_{21}^l & w_{22}^l & \dots & w_{2n_l}^l \\ \vdots & \vdots & & \vdots \\ w_{n_{l-1}1}^l & w_{n_{l-1}2}^l & \dots & w_{n_{l-1}n_l}^l \end{bmatrix},$$

$$A^l = \begin{bmatrix} a_1^l \\ a_2^l \\ \vdots \\ a_{n_l}^l \end{bmatrix}, \qquad B^l = \begin{bmatrix} b_1^l \\ b_2^l \\ \vdots \\ b_{n_l}^l \end{bmatrix}, \qquad Z^l = \begin{bmatrix} z_1^l \\ z_2^l \\ \vdots \\ z_{n_l}^l \end{bmatrix},$$

such that we get

$$Z^l = (W^l)^T A^{l-1} + B^l, \qquad A^l = \sigma(Z^l). \tag{2.2}$$

The complete feed forward algorithm is simply done by using equation 2.2 for layer $l = 2, \dots, L$, for which we produce the NN prediction $\vec{y}$.

Back Propagation

The back propagation algorithm is the essential algorithm behind the training of the neural network. This serves the purpose of tuning the weights and biases according to a given cost function. A common choice for regression type problems is the mean squared error

$$C(\vec{y}) = ||\vec{y} - \vec{t}||_2^2 = \frac{1}{N} \sum_{i=1}^{N} (y_i - t_i)^2, \tag{2.3}$$

where N is the number of data points. However, one can choose from a wide selection of cost functions in general. For classification cases it is normal to use Cross entropy as the cost function. In that case it is defined as

$$C(\vec{y}) = -\sum_{i=1}^{n} \Big[y_i \log(t_i) + (1 - y_i) \log(1 - t_i) \Big]. \tag{2.4}$$

By calculating the gradient gradient $\nabla_{w,b} C$ with respect to the weights and biases, we can use gradient descent (see section **??**) to minimize the cost function. Thus we need to evaluate $\partial C / \partial w_{ij}^l$ and $\partial C / \partial b_j^l$. We begin by looking at the last layer $l = L$. By using the chain rule we get

$$\frac{\partial C}{\partial w_{ij}^L} = \frac{\partial C}{\partial a_j^L} \frac{\partial a_j^L}{\partial z_j^L} \frac{\partial z_j^L}{\partial w_{ij}^L}.$$

We remember that

$$a_j^L = \sigma(z_j^L), \qquad\qquad z_j^L = \sum_{i=1}^{n_{L-1}} w_{ij}^L a_i^{L-1} + b_j^L,$$

such that we get

$$\frac{\partial C}{\partial w_{ij}^L} = \frac{\partial C}{\partial a_j^L} \sigma'(z_j^L) a_i^{L-1}.$$

Notice that the remaining derivatives can easily be calculated when deciding on a specific cost and activation function. For the bias we simply get

$$\frac{\partial C}{\partial b_j^L} = \frac{\partial C}{\partial a_j^L} \frac{\partial a_j^L}{\partial z_j^L} \frac{\partial z_j^L}{\partial b_j^L} = \frac{\partial C}{\partial a_j^L} \sigma'(z_j^L) \cdot 1.$$

We use this to motivate the definition of the local gradient

$$\delta_j^l \equiv \frac{\partial C}{\partial z_j^l},$$

such that

$$\delta_j^L = \frac{\partial C}{\partial z_j^L} = \frac{\partial C}{\partial a_j^L} \frac{\partial a_j^L}{\partial z_j^L} = \frac{\partial C}{\partial a_j^L} \sigma'(z_j^L).$$

This yields the more compact expressions

$$\frac{\partial C}{\partial w_{ij}^L} = \delta_j^L a_i^{L-1}, \qquad\qquad \frac{\partial C}{\partial b_j^L} = \delta_j^L.$$

The local gradient δ_j^l is also commonly called the "error" since it reflects how big of an influence the j^{th} node in layer l have on the change of the cost function. We let $\boldsymbol{\delta}^l$ denote the vector containing all the local gradients associated with layer l and we can write it out as a matrix equation for the last layer

$$\boldsymbol{\delta}^L = \nabla_a C \odot \frac{\partial \sigma}{\partial z^L}, \quad \nabla_a C = \left[\frac{\partial C}{\partial a_1^L}, \frac{\partial C}{\partial a_2^L}, \ldots, \frac{\partial C}{\partial a_{n_L}^L} \right]^T,$$

where $\odot$ is the Hadamard product (element-wise product). We can then define the gradient δ_j^l for the j^{th} node on a general layer l in terms of δ_k^{l+1} for the k^{th} node on the following layer $l + 1$ by using the chain rule

$$\begin{aligned} \delta_j^l &\equiv \frac{\partial C}{\partial z_j^l} \\ &= \sum_k \frac{\partial C}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l} \\ &= \sum_k \frac{\partial z_k^{l+1}}{\partial z_j^l} \delta_k^{l+1}. \end{aligned} \tag{2.5}$$

We remember

$$z_k^{l+1} = \sum_{j=1}^{n_l} w_{jk}^{l+1} a_j^l + b_k^{l+1} = \sum_{j=1}^{n_l} w_{jk}^{l+1} \sigma(z_j^l) + b_k^{l+1},$$

and by taking the derivative we obtain

$$\frac{\partial z_k^{l+1}}{\partial z_j^l} = w_{jk}^{l+1} \sigma'(z_j^l). \tag{2.6}$$

We substitute back 2.6 into 2.5 to find

$$\delta_j^l = \sum_k \delta_k^{l+1} w_{ij}^{l+1} \sigma'(z_j^l).$$

The complete back propagation algorithm then becomes

- Compute

$$\delta_j^L = \frac{\partial C}{\partial a_j^L} \sigma'(z_j^L).$$

- For $l = L-1, L-2, \ldots, 1$ compute:

$$\delta_j^l = \sum_k \delta_k^{l+1} w_{jk}^{l+1} \sigma'(z_j^l).$$

- For all layers l update weights and biases:

$$w_{ij}^l \leftarrow w_{ij}^l - \eta \delta_j^l a_i^{l-1},$$
$$b_j^l \leftarrow b_j^l - \eta \delta_j^l,$$

where η is the learning rate.

In order to minimize the risk of overfitting it is common to include a L2 regularization by adding the L2 norm to the cost function as $\lambda||w||_2^2$ and $\lambda||b||_2^2$. This result in the modified expressions for the update

$$w_{ij}^l \leftarrow w_{ij}^l - \eta(\delta_j^l a_i^{l-1} + \lambda w_{ij}^l),$$
$$b_j^l \leftarrow b_j^l - \eta(\delta_j^l + \lambda b_j^l).$$

2.1.3 Gradient Boosting and decision trees

In this rapport I will use the XGBoost-classifier which uses gradient-boosted trees. Gradient-boosting is a machine learning algorithm which uses a collective of "weak" classifiers in order to create one strong classifier. In the case of gradient-boosted trees the weak classifiers are a collective of shallow trees, which combine to form a classifiers that allows for deeper learning. As is the case for most gradient-boosting techniques, the collecting of weak classifiers is an iterative process.
We define an imperfect model $\mathcal{F}_m$, which is a collective of m number of weak classifiers, estimators. A prediction for the model on a given data-points, x_i is defined as $\mathcal{F}_m(x_i)$, and the observed value for the aforementioned data is defined as y_i. The goal of the iterative process is to minimize some cost-function $\mathcal{C}$ by introducing a new estimator h_m to compensate for any error, $\mathcal{C}(\mathcal{F}_m(x_i), y_i)$. In other words we define the new estimator as:

$$\tilde{\mathcal{C}}(\mathcal{F}_m(x_i), y_i) = h_m(x_i), \tag{2.7}$$

where we define $\tilde{\mathcal{C}}$ as some relation defined between the observed and predicted values such that when added to the initial prediction we minimize $\mathcal{C}$.
Using our new estimator h_m, we can now define a new model as

$$\mathcal{F}_{m+1}(x_i) = \mathcal{F}_m + h_m(x_i). \tag{2.8}$$

The XGBoost [5] framework used in this analysis enables a gradient-boosted algorithm, and was initially created for the Higgs ML challenge. Since the challenge, XGBoost has become a favorite for many in the ML community and has later won many other ML challenges. XGBoost often outperforms ordinary decision trees, but what is gains in results it looses in interpretability. A single tree can easily be analysed and dissected, but when the number of trees increases this becomes harder.

2.2 Statistical and multi-variabel analysis

The main idea of the search is to define a region where we will compare the Monte Carlo (MC)-background to the data and analyse any differences. Any analysis in the signal region hopes to either discovery or exclude certain Beyond Standard Model (BSM) physics. The standatd way of doing so is through statistics. In this rapport the statistical analysis will not be the focus and will therefore not be explained in heavy detail. Nonetheless a high-level understanding of the statistics is neccesarry when discussing the final results results. Therefore I will in this section discuss and define some basic statistical expressions and formulas.

Chapter 3

Implementation

3.1 Tools and Data

Every year technology for generating and measuring particle collisions is improved. As a consequence, the amount of data increases drastically. The ATLAS experiment is one of the largest particle detector experiments currently operating at the CERN laboratory near Geneva. ATLAS alone generates approximately 1 petabyte of raw data every second from proton-proton collisions at the Large Hadron Collider (LHC). With amounts of data this large, data handling and storing is a big challenge. Therefore, taking advantage of sophisticated numerical tools and data frameworks is pivotal if scientific development is to keep up with technological development.
In this section I will cover some tools and frameworks I have used to complete my analysis. Large amounts of details and explanations will not be covered. Instead, this section will highlight which tools were used and some motivation for choosing them. Additionally, I will cover some details regarding the data being used, both MC and real.

3.1.1 Monte Carlo Data

3.1.2 ROOT, RDataframe and Pandas

ROOT [6] is at its core a large $C++$ library and data structure made specifically for big data analysis and computing, as well as data visualization. Today, all ATLAS data is stored as a ROOT-file along with more than 1 exabyte of data worldwide. ROOT has many High Performance Computing (HPC) qualities which makes it ideal for particle physics analysis which demands heavy computations. Additionally, many particle physics-specific packages have been developed to make it an even better tool. Any function not already in library, can easily be added in a HPC-effective manner through $C++$.
All distribution plots made for this thesis were created using ROOT. ROOT has implemented a highly intuitive and effective Application Programming Interface (API) for data comparison and visualization. ROOT allows for quick and direct comparison between data through an advanced graphical user interface. Additionally, a lot of functionality for creating complex stacked histogram are implemented in the ROOT API, such as uncertainty calculations and ordering of histograms.
As for the data structure, the raw data was loaded as a ROOT-file. To easier handle the data and add new features, I used the RDataFrame structure [7]. RDataFrame allows for easy addition of new columns as well as filtering of events through native functionality. As a consequence I used RDataFrame to calculate all higher-level features such as M_{T2} and M_{lll}. An example is shown in the following section.
After all data handling was complete, I used ROOT's *AsNumpy* function to translate the data frame as a numpy object, which then allowed me to transform it to a Pandas-data frame [8]. This is done because Pandas, like most ML-tools work in a strict Python environment. Pandas, similarly to RDataFrame includes many deep computational libraries, and is optimal for analysis of big data. When all ML is completed, the data is transformed back to RDataFrame, to take advantage of plotting functionality in ROOT.

3.1.3 Computing features in ROOT: Example

In this section I will cover a simple example to highlight the steps taken in generating the dataset I used during analysis. As mentioned earlier, the two main frameworks used were RDataFrame and Pandas. In this example I will cover the case of a feature not already in the ROOT file, namely the trilepton invariant mass. All loading of data is done using the ROOT framework and is quickly made into a RDataFrame. To effectively generate new features, we want to stay in ROOT and RDataFrame. Therefore, we create our $C++$-file, *helperfunction*. The helperfunction contains all additional ROOT function that are used in the analysis and is not already native to ROOT. In the case of computing the trilepton invariant mass, the $C++$-function is created like shown in 3.1. In 3.1, we see a couple of measures taken to uphold to the ROOT environment. The first is the typecasting to $VecF\_t$. $VecF\_t$ is created to wrap floats in the native ROOT vector object, $RVec$. The same is done in other cases such as float and integers, $VecI\_t$ and $VecB\_t$. The second measure was using $TLorentzVector$ [9] to calculate the invariant mass. $TLorentzVector$ is a class native to ROOT with many built-in functions. In this case we use the class to create three vectors through the variables, P_t, η, ϕ and M. Then, through the $TLorentzVector$ class we can simply add all three vector together, and extract the invariant mass.

```
// Compute the trilepton invariant mass
float ComputeInvariantMass(VecF_t& pt, VecF_t& eta, VecF_t& phi, VecF_t& m) {
  TLorentzVector p1;
  TLorentzVector p2;
  TLorentzVector p3;
  p1.SetPtEtaPhiM(pt[0], eta[0], phi[0], m[0]);
  p2.SetPtEtaPhiM(pt[1], eta[1], phi[1], m[1]);
  p3.SetPtEtaPhiM(pt[2], eta[2], phi[2], m[2]);
  return (p1 + p2 + p3).M();
}
```

Listing 3.1: $C++$-function for M_{lll}.

With a helper function created in $C++$ we can move over to a Python and RDataFrame environment for calculation and plotting. In the code written in 3.2, I have shown a simple example of loading new $C++$-functions, filtering out bad events, calculating new features and adding said features to a histogram. The first three lines of code both process and include the helperfunction functions into the ROOT framework. Then I loop over all keys in the data frame, which in my case are the different channels (i.e Diboson, $t\bar{t}$ etc.). For each channel I filter out 'good' events, based on the criteria from section 3.2.1. Then, I use RDataFrame's $Define$ function to calculate and add a new feature using our $ComputeInvariantMass$-function. Finally, I save the feature as an $Histo1D$ object, which I plot later using ROOT plotting API's.

```
R.gROOT.ProcessLine(".L helperFunctions.cxx+");
R.gInterpreter.Declare('#include "helperFunctions.h"')
R.gSystem.Load("helperFunctions_cxx.so")

for k in df.keys():
    # Define good leptons
    isGoodLepton = "feature1 < cut1 && feature2 >= cut2"

    # Define good leptons in dataframe
    df[k] = df[k].Define("isGoodLepton",isGoodLepton)

    # Define number of good leptons
    df[k] = df[k].Define("nGoodLeptons","ROOT::VecOps::Sum(isGoodLepton)")

    # Demand 3 good leptons
    df[k] = df[k].Filter("nGoodLeptons == 3")

    # Define Invariant Mass (lll)
    df[k] = df[k].Define("mlll","ComputeInvariantMass(lepPt[isGoodLepton],
                                                      lepEta[isGoodLepton],
                                                      lepPhi[isGoodLepton],
                                                      lepM[isGoodLepton])")
    # Add to histogram
    histo["mlll_%s"%k] = df[k].Histo1D(("mlll_%s"%k,
                                        "mlll_%s"%k,40,50,500),
                                        "mlll",
                                        "wgt_SG")
```

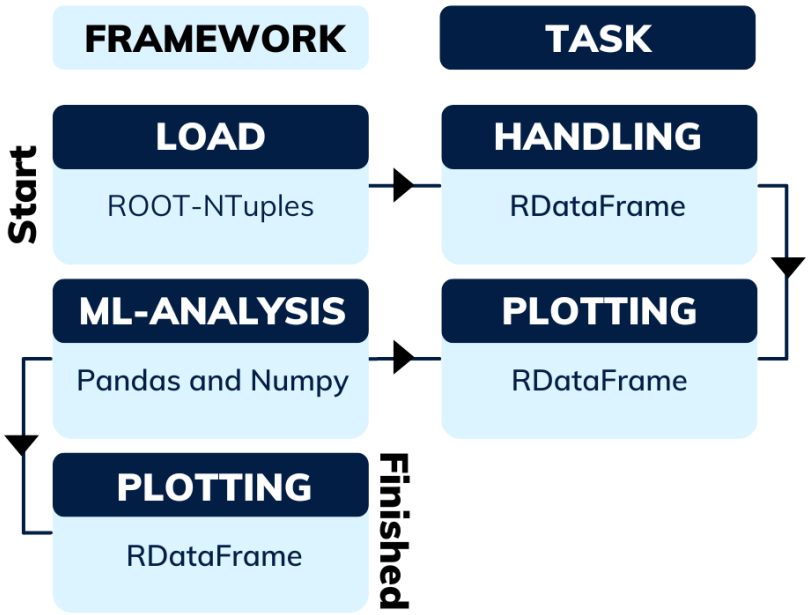


Figure 3.1: A visual summary of the workflow and framework use for the computational analysis.

Listing 3.2: Python-file for calling dataframe and calculating M_{lll}.

In this example I chose the trilepton invariant mass, but in the analysis all high-level features were calculated using a similar method. The workflow of the method is visualized in figure 3.1. We load data in ROOT format, data handling and plot feature distrobution in RDataFrame, perform ML- analysis in Pandas and plot results in RDataFrame.

3.2 Features

The choice of which features to study and which to neglect are crucial in a search for new physics. This is particularly true in the case of applying machine learning. The general motivation for including a given feature can be based on several factors. The first being its ability to provide a trend which we as researches can exploit when creating our regions. By this I mean that it is a variable were there is diversity in distribution between the different channels. The second motivation is grounded in physics. Often we as physicists tend to lean towards variables we know have some effect one the physics we are studying. For example the variable E_T^{miss}, can be directly used to either include or discard events were we do or do not expect final states with sufficient missing energy. The final motivation is grounded in the MC-simmulations ability to represent the variable. If there seems to be a clear deviation between the real and MC-data which does not stem from any new physics, we tend to discard them from the analysis.

3.2.1 Cuts and triggers

To allow for deep learning and a thorough analysis one must try and keep as much of the data as possible. At the same time, including large amounts of irrelevant data can be both redundant and destructive. Therefore, simple cuts are necessary. The cuts applied in the analysis were grouped in two definitions, baseline and signal. The baseline requirements are written in table 3.1 and the signal requirements are written in table 3.2. Both sets of requirments were taken from tha ATLAS article from 2022 [10]. Given the definitions we demand that each event contains exactly three signal and three baseline leptons, thereby removing any event with more or less.

Requirement	Baseline electrons	Baseline muons
Identification		Loose
Overlap Removal	lepPassOR	lepPassOR
$\eta-$ cut	$\lvert\eta\rvert < 2.47$	$\lvert\eta\rvert < 2.7$
$\lvert z_0 \sin(\theta)\rvert$ cut	$\lvert z_0 \sin(\theta)\rvert < 0.5$ mm	$\lvert z_0 \sin(\theta)\rvert < 0.5$ mm

Table 3.1: Requirments for baseline electrons and muons.

Requirement	Signal electrons	Signal muons
Baseline	yes	yes
Identification	Tight	
Isolation	LooseVarRad	LooseVarRad
$\lvert d_0\rvert/\sigma_{d_0}$ cut	$\lvert d_0\rvert/\sigma_{d_0} < 5.0$	$\lvert d_0\rvert/\sigma_{d_0} < 3.0$

Table 3.2: Requirments for signal electrons and muons.

Leptons are identified in the detector by using a likelihood-based method combining information from different parts of the detector. The criteria of Loose or Tight identification are simply different thresholds in the discriminant, where Loose is defined as a lower threshold than Tight [11]. The overlap removal is used to solve any cases where the same lepton has been reconstructed as both a muon and an electron. The boolean of *lepPassOr* simply applies a set of requirements to avoid any double counting. The cut for the longitudinal track parameters, z_0 is applied to ensure that the leptons originate for the primary vertex.
As for the requirements for the signal leptons, we require all baseline requirements are passed with the addition of a few more. We require Loose isolation for both electrons and muons. This means requiring criteria for a cone around the lepton and is used to suppress QCD-background events. Similarly to the z_0-cut, the transverse track parameter is also used to ensure origin from primary vertex.

In addition to the simple cuts, we must insure a good comparison between MC- and real data. Often one finds large deviation between the two in the case of either very large or very small P_t. The latter case can often be caused by poor reconstruction or miss identification. These are issues we aim to solve by checking different triggers. Given our data set is composed of different data sets spread over many years, different triggers are used.

3.2.2 Lepton variables selection

Now we will have a look at what variables from the leptons that were included in the analysis. All low level information on the momentum of the leptons were added into the dataset: i.e the transverse momentum P_t, the pseudo rapidity η and the azimuthal angle ϕ. All momentum features were represented individually for each lepton. For example P_t was added as three columns, $P_t(l_1)$, $P_t(l_2)$ and $P_t(l_3)$, where the ordering of the leptons were based on the momentum from highest (l_1) to lowest (l_3). Similarly I added information regarding the charge ($\pm$) and flavor (electron, muon) of each lepton. Based on the momentum variables the transverse mass m_t of each lepoton was calculated and included along with the energy E_t^{miss} and azimuthal angle ϕ^{miss} of the missing transverse momentum.
The variables described in the section above are often considered as low-level features. These are very useful in many (if not all) searches and contribute little to no bias to your analysis. But, in the case of final-state spesific searches such as mine, one can allow one self of adding physics motivated higher-level physics. The higher level features calculated in this thesis were inspired by [10] (ATLAS 2022).
Firstly I added different mass variables, namely m_{lll} and $m_{ll}(OSSF)$. The first being the trilepton invariant mass

2015	2016	2017 + 2018
HLT_2e12_lhloose_L12EM10VH	HLT_2e17_lhvloose_nod0	HLT_2e17_lhvloose_nod0_L12EM15VHI
HLT_e17_lhloose_mu14	HLT_e17_lhloose_nod0_mu14	HLT_e17_lhloose_nod0_mu14
HLT_mu18_mu8noL1	HLT_mu22_mu8noL1	HLT_mu22_mu8noL1
		HLT_2e24_lhvloose_nod0

Table 3.3: Trigger requirments for events produced in their respective years.

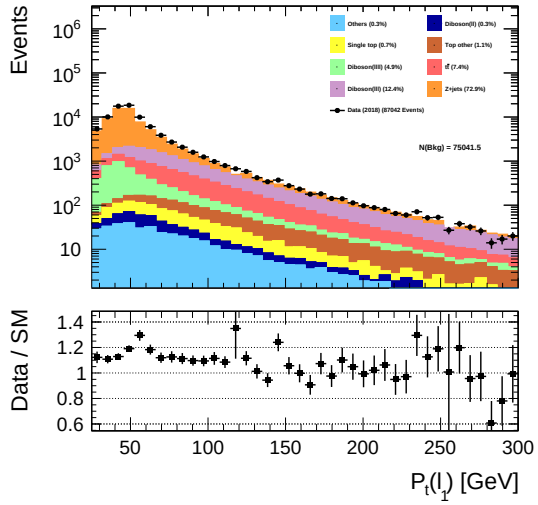
and the latter being the dilepton invariant mass of the pair with Opposite Sign Same Flavour (OSSF). In the case of more than one possible OSSF-pair, the pair with the highest invariant mass was chosen. Secondly I added variables composed of the sum of different set of momentum. These variables are the sum of all three leptons $H_t(lll)$, of the pair with Same Sign (SS) $H_t(SS)$ and the sum of the momentum for all three leptons added with the missing transverse energy $H_t(lll) + E_t^{miss}$. Finally I added the significance of the E_t^{miss}, $S(E_t^{miss})$.

3.2.3 Jet variables selection

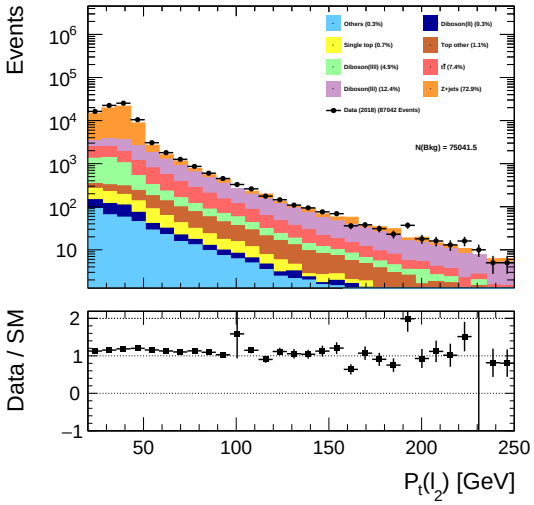
Now we can have a look at the jet-features. Given the final-state of interest should be independant of jets, there are not many features added with jet information. But, given the risk of missidentification and errors in reconstruction, some features were added. The first features were the number of jets, both all signal jets and number of b-jets. The latter information was divided in two columns based on the efficency of a multivarant analysis used to sepereata jet-flavors. The effeciencies used are 77% and %85. The last information added for the jets were the mass of the leading pair (based on p_t) di-jet mass.

3.2.4 Validation

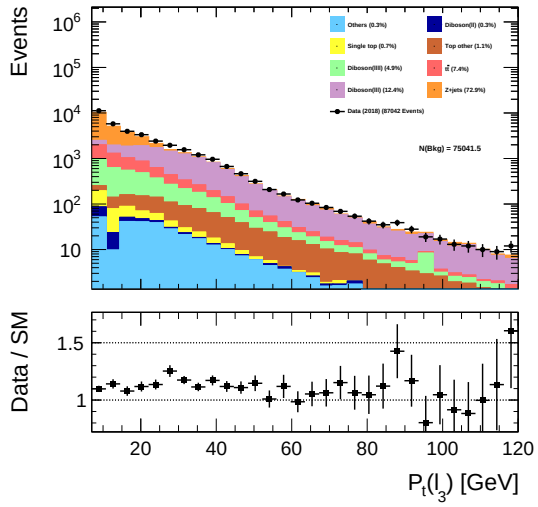
As mentioned in previous sections, the comparison between ML- and real data is a crucial part of the analysis, and we must therefore insure an adequate reconstruction of the real data. This is not only true for the low-level features taken directly from the ML simulation, but also for the the higher-level features. Therefore we will in this section compare both sets of data for all features included in the anlysis.



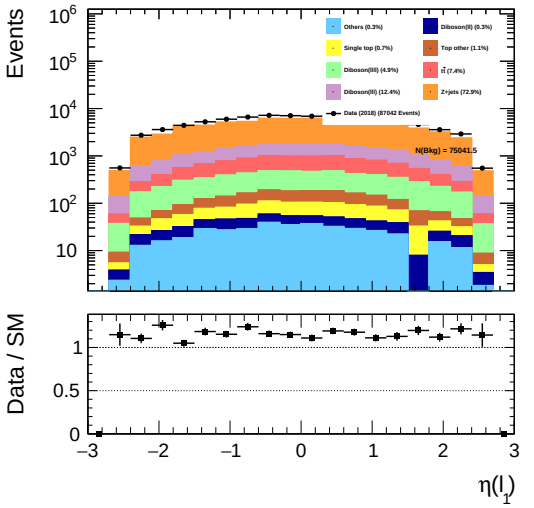
(a)



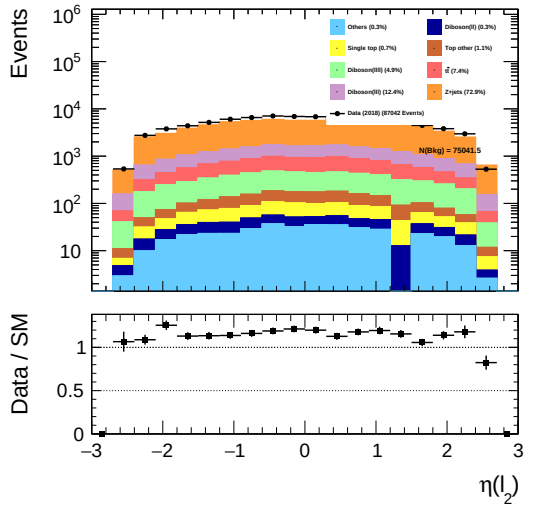
(b)



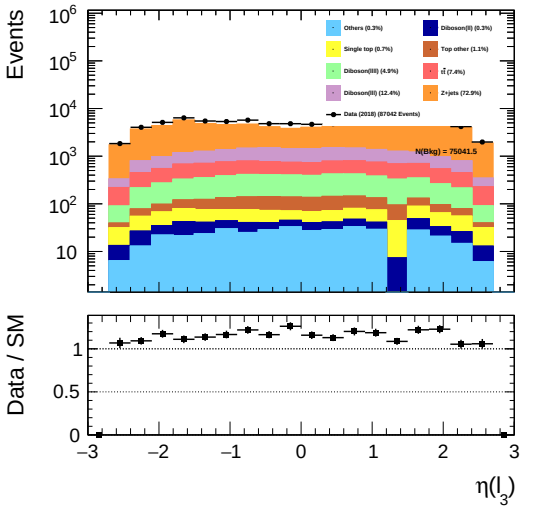
(c)



(d)



(e)



(f)

Figure 3.2: The event distribution for for each channel over P_t for the first 3.2a , second 3.2b and third 3.2c lepton. Similarly the distribution over η for the 3.2d, second 3.2e and third 3.2f lepton

3.2.5 Negative Weights

The negative weight problem is a well known problem in the world of ML-analysis for HEP, and is a consequence of higher perturbative accuracy. For the purpose of visualizing distributions, negative weights are not a problem. But, when using the weights in the training of the classifier, the XGBoost framework will not work.
Before deciding how to mediate the issue, I decided to plot the distribution for a small subset of features for all the events with negative weights.
In figure 3.8a I plotted the absolute value P_t distribution of events with negative weights and compare it to all the events in figure 3.8b. Similarly I have plotted the same values for the second lepton in figures 3.8b and 3.8d respectively. From the two comparisons we see that the distribution of negative weights is relatively equal to that of all the events. The same comparisons have been made in figures 3.9a, 3.9b, 3.9d and 3.9d respectively but for ϕ. The same observations are made for both P_t and ϕ. This means we can be justified in treating the negative weight distribution uniformally across the feature space.

How one chooses to deal with this problem can heavily effect results and many solutions are suggested as a consequence. Given the time main focus of this rapport, the simplest solution was chosen. Namely, I chose to normalize all the weights as a whole, conserving the total sum of the weights and at the same time changing all negative signs. The simple procedure is shown bellow in equation 3.1,

$$w_i = P \mid w_i \mid \tag{3.1}$$

$$P = \sum_{j=0}^{N-1} \frac{w_j}{\mid w_j \mid}, \tag{3.2}$$

where w_i is the weight of an event i, $i \in [0, N-1]$ and N equals the total number of events. Although this is a very simplistic solution, our observations made for the distribution of negative weights can be used to justify the approach.

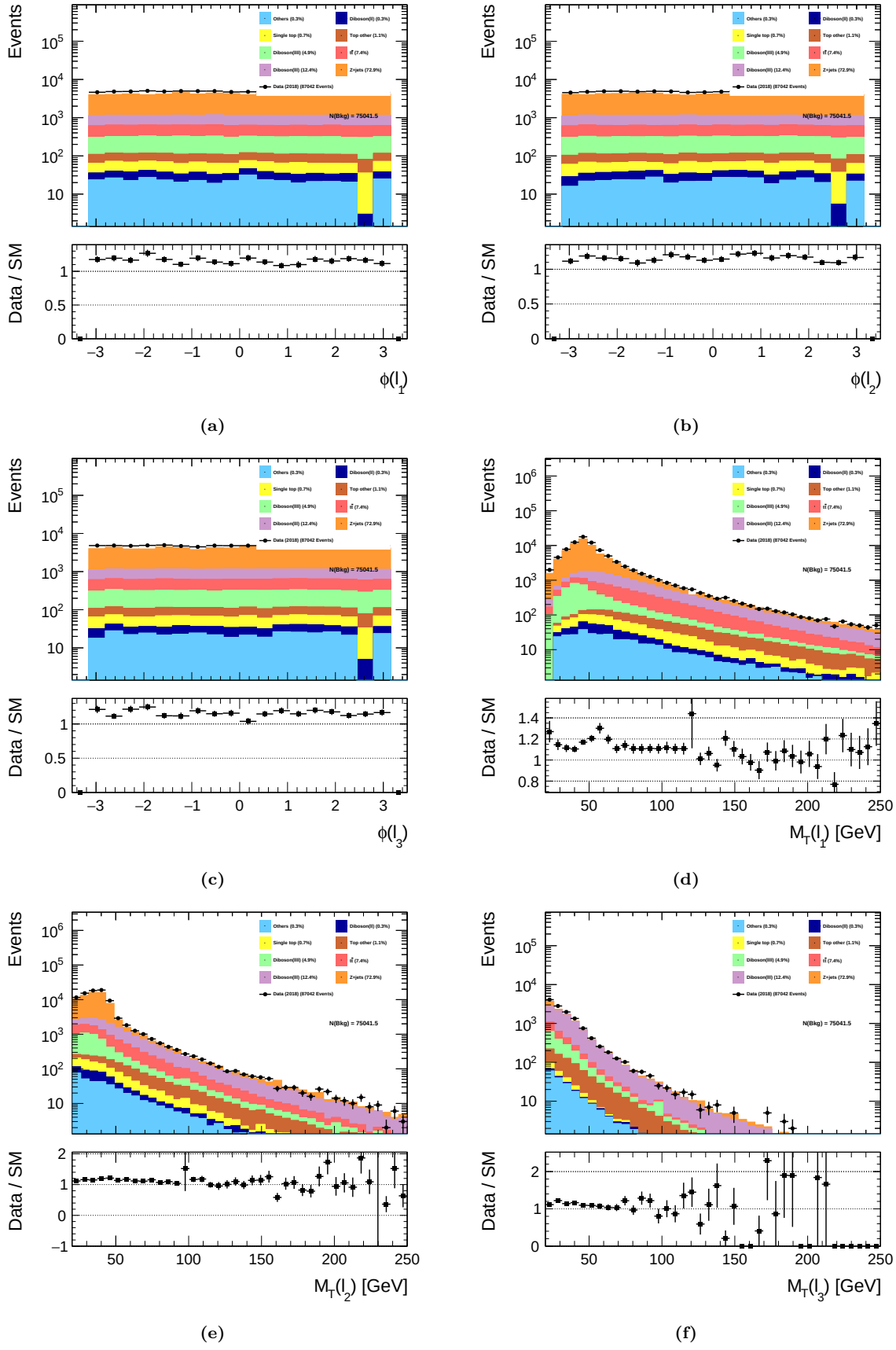
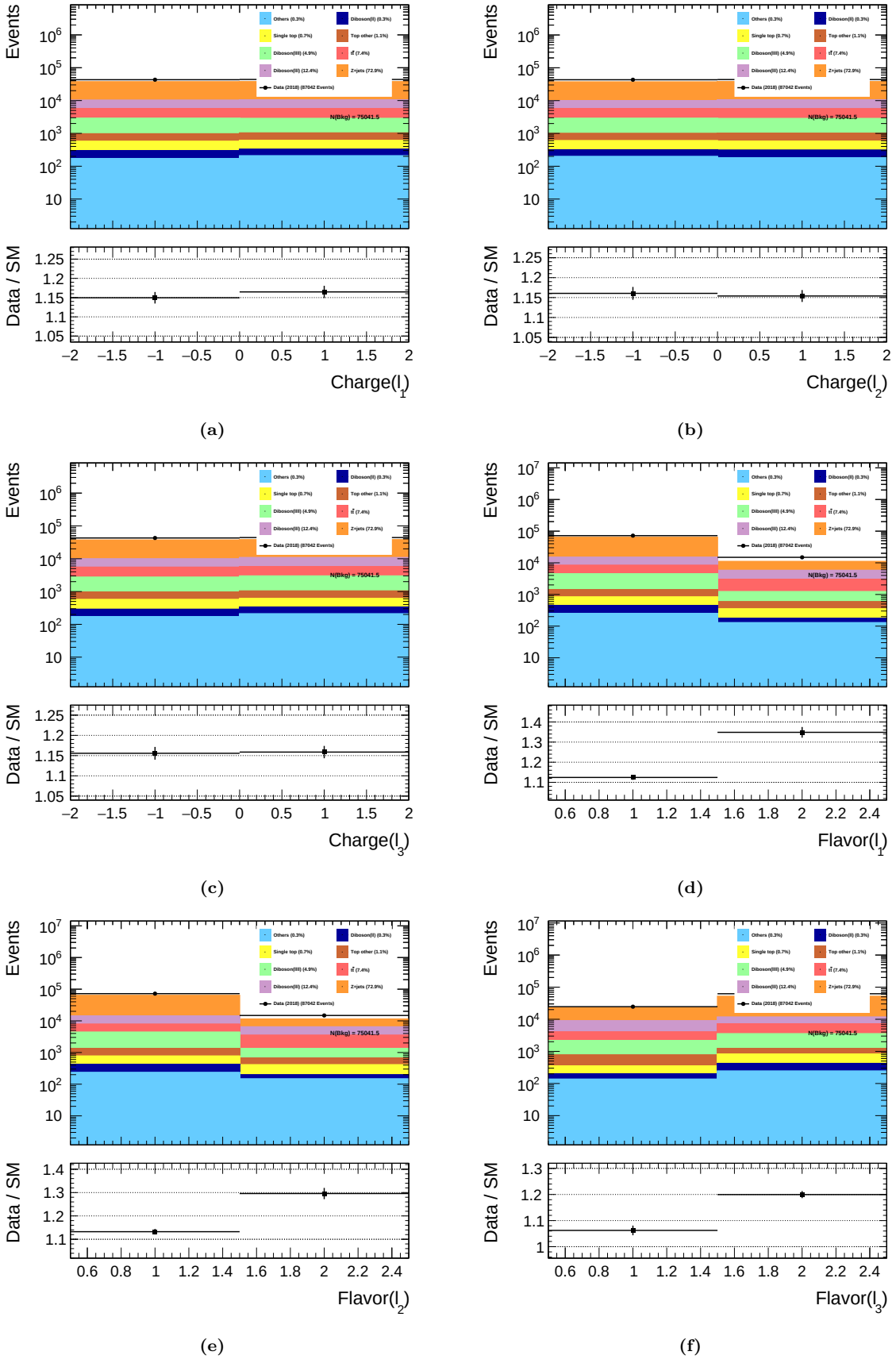
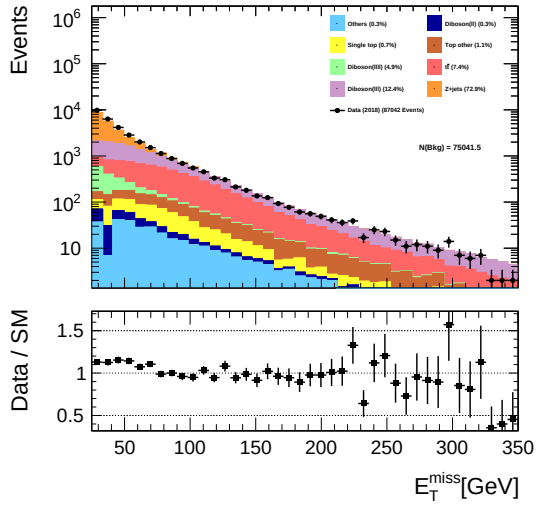


Figure 3.3: The event distribution for for each channel over ϕ for the first 3.3a , second 3.3b and third 3.3c lepton. Similarly the distribution over m_t for the first 3.3d, second 3.3e and third 3.3f lepton

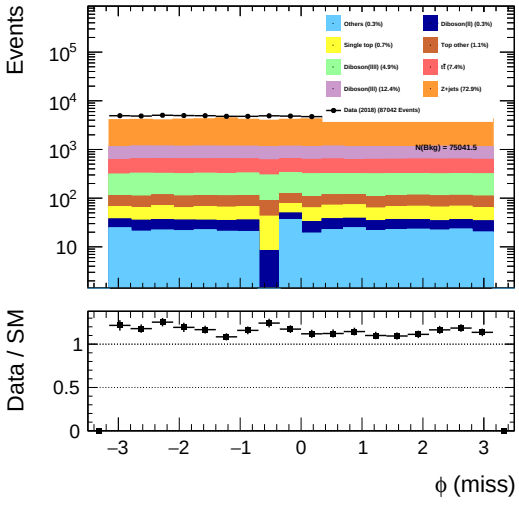


(a) **(b)** **(c)** **(d)** **(e)** **(f)**

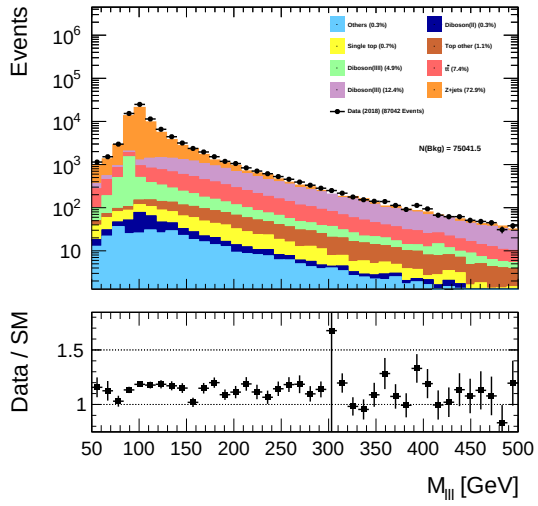
Figure 3.4: The event distribution for for each channel over the charge for the first 3.4a , second 3.4b and third 3.4c lepton. Similarly the distribution over the flavor for the first 3.4d, second 3.4e and third 3.4f lepton



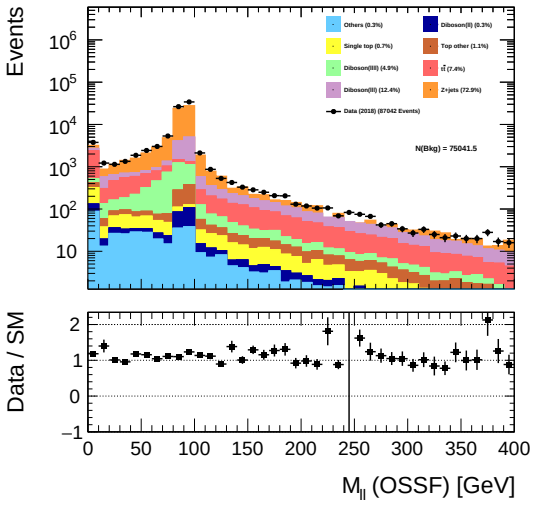
(a)



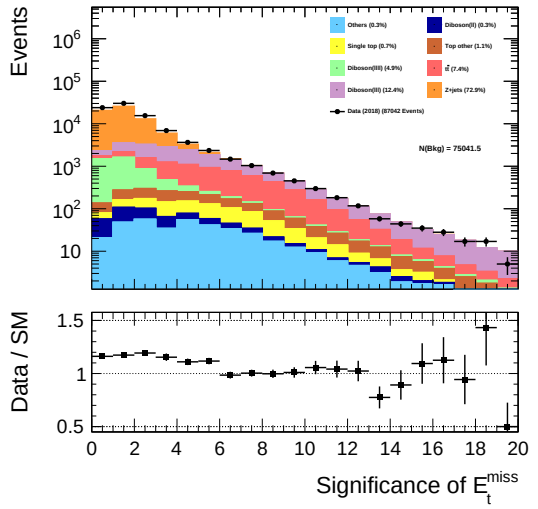
(b)



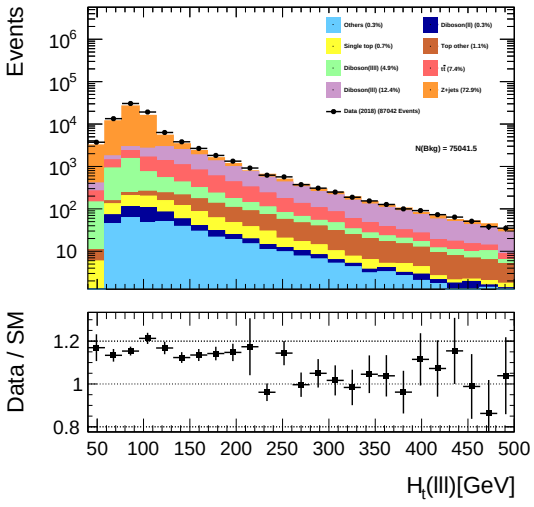
(c)



(d)

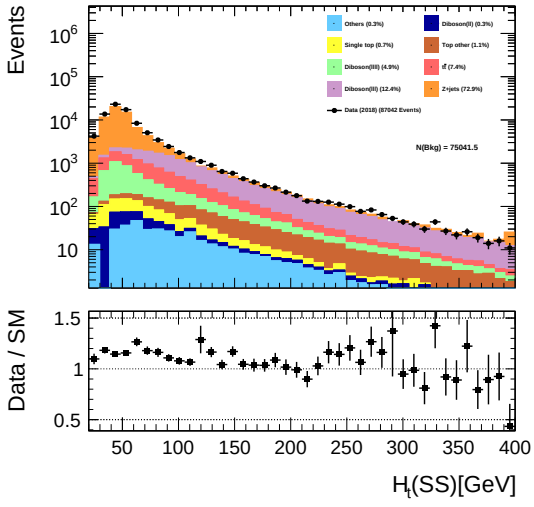


(e)

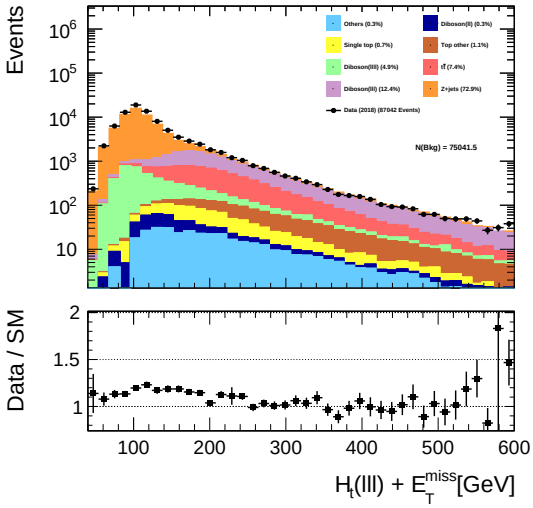


(f)

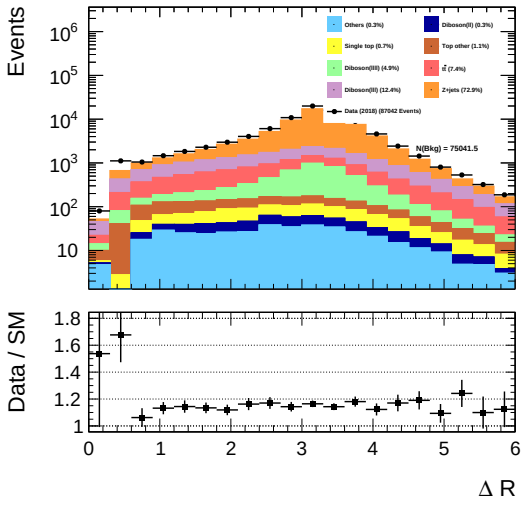
Figure 3.5: The event distribution for for each channel over the energy 3.5a and azimuthal angel 3.5b for the transverse momentum. The distribution of the invariant mass of the three leptons 3.5c and the OSSF pair 3.5d. The distribution over the significance of the missing transverse enegy 3.5e and the sum of P_t 3.5f.



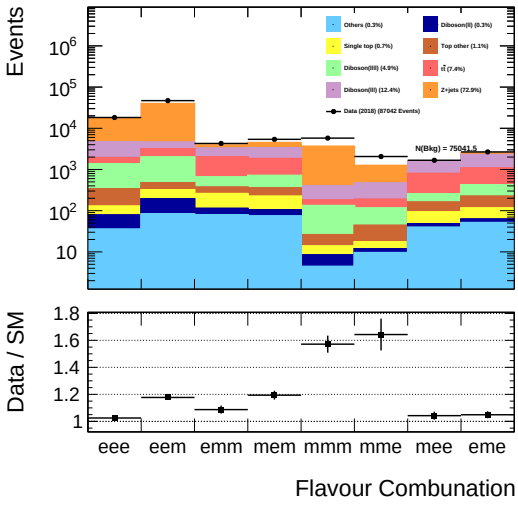
(a)



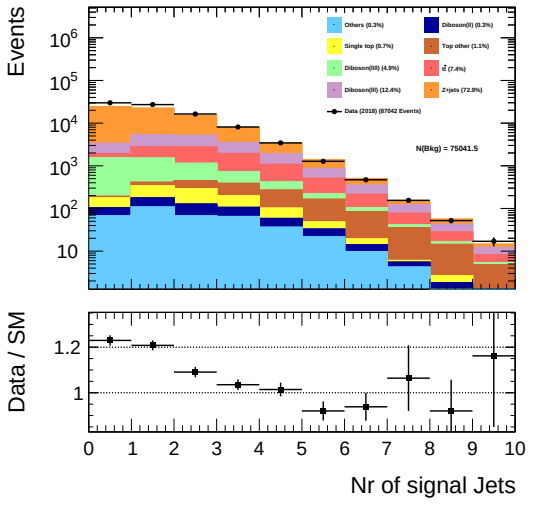
(b)



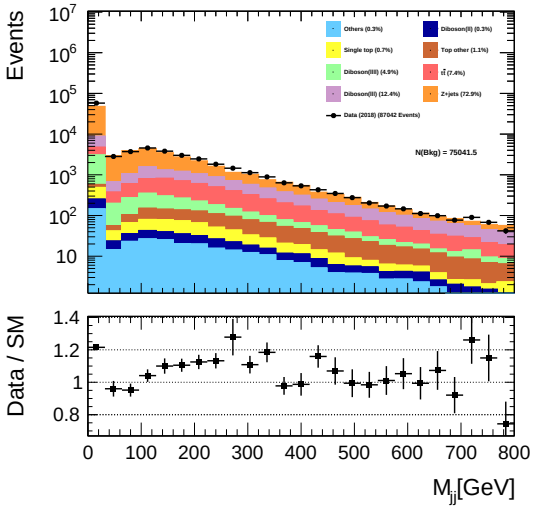
(c)



(d)

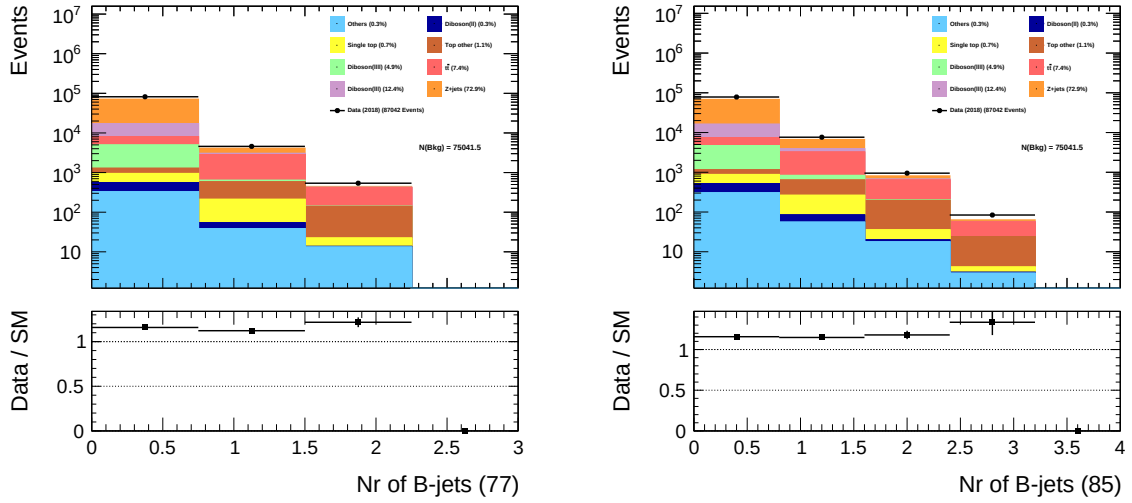


(e)



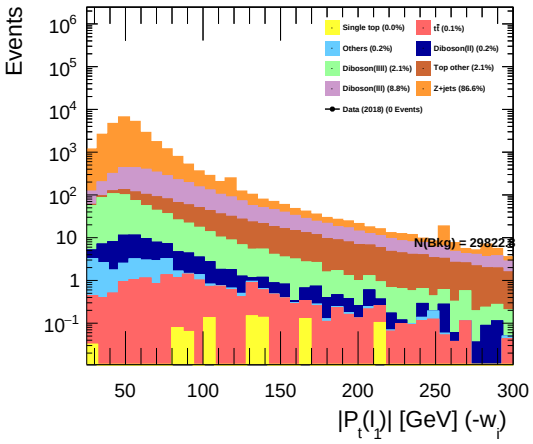
(f)

Figure 3.6: The event distribution for for each channel over the sum of P_t for the SS pair 3.6a and the sum over all three leptons added with E_t^{miss} 3.6b. The distribution over ΔR 3.6c and the flavor combination of the three leptons 3.6d. The distribution of number of jets 3.6e and the mass of the leading di-jet pair 3.6f.

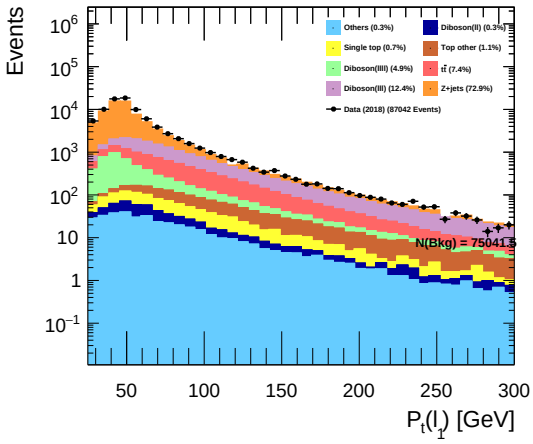


(a) **(b)**

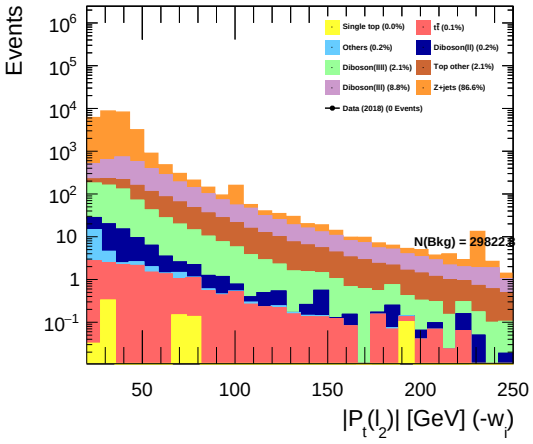
Figure 3.7: The event distribution for for each channel over the number of b-jets with 77% 3.7a and 85% 3.7b efficency.



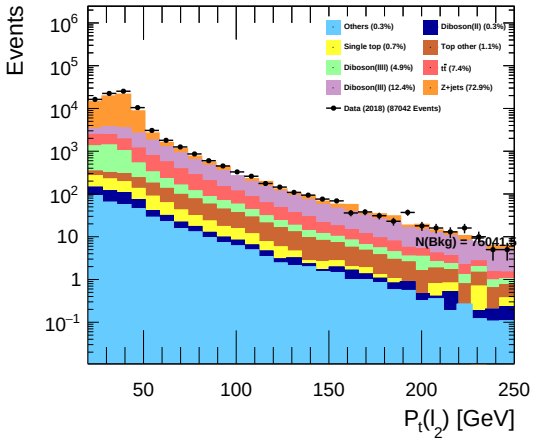
(a)



(b)

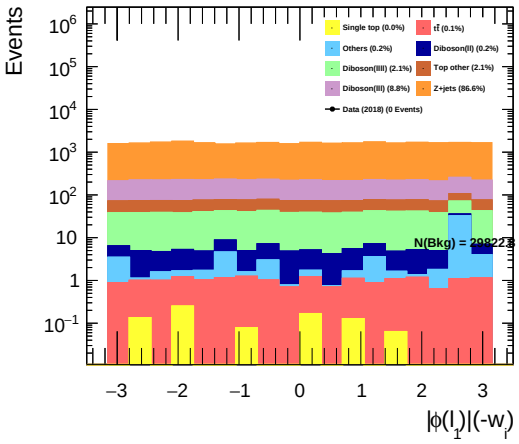


(c)

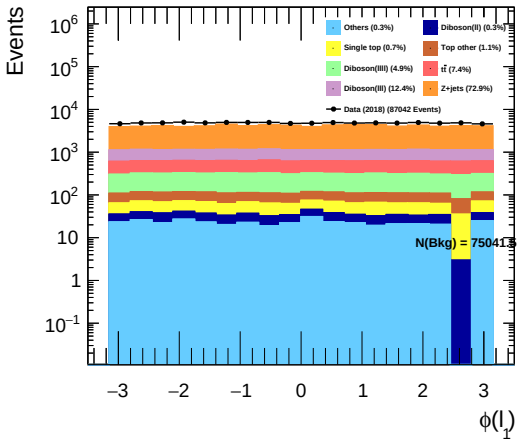


(d)

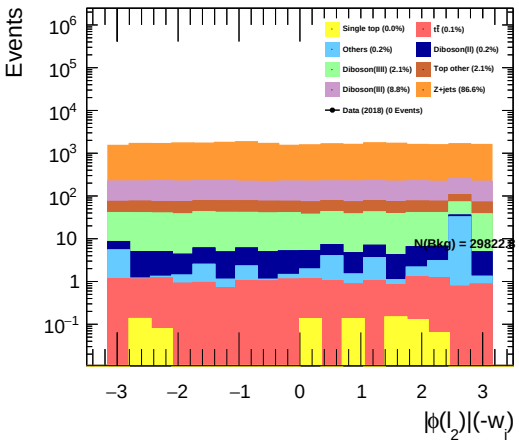
Figure 3.8: The absolute value of the P_t for events with stricly negative weights (l_1 3.8a, l_2 3.8c) and all events (l_1 3.8b, l_2 3.8d).



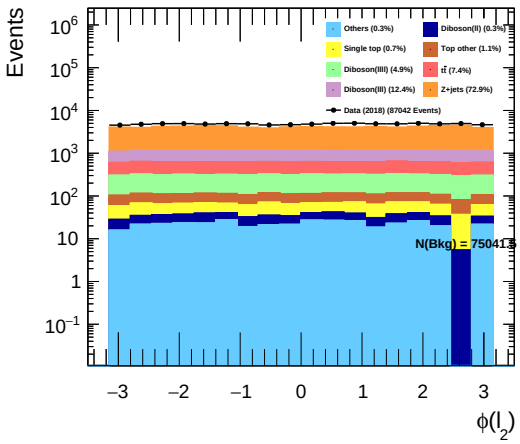
(a)



(b)



(c)



(d)

Figure 3.9: The absolute value of the ϕ for events with strictly negative weights (l_1 3.9a, l_2 3.9c) and all events (l_1 3.9b, l_2 3.9d).

Appendices

Appendix A

Acronyms

API Application Programming Interface

BSM Beyond Standard Model

CP Charge-Parity

DNN Deep Neural Networks

HPC High Performance Computing

LHC Large Hadron Collider

MC Monte Carlo

ML Machine Learning

NN Neural Network

OSSF Opposite Sign Same Flavour

QCD Quantum Chromo Dynamics

QED Quantum Electro Dynamics

SM Standard Model

SR Signal region

SS Same Sign

Bibliography

[1] J. Joyce, *Finnegans Wake*. Penguin Books, New York, 1999.

[2] W. Thomson, *Lord kelvin addresed the british association for the advancement of science*, 1900.

[3] D. Guest, J. Collado, P. Baldi, S.-C. Hsu, G. Urban and D. Whiteson, *Jet flavor classification in high-energy physics with deep neural networks*, *Physical Review D* **94** (dec, 2016) .

[4] J. Pumplin, *How to tell quark jets from gluon jets*, *Phys. Rev. D* **44** (Oct, 1991) 2025–2032.

[5] The XGBoost Contributors, "Xgboost."

[6] The Root team, "Root."

[7] The Root team, "Rdataframe."

[8] The Pandas team and voluntary contributers, "Pandas."

[9] The Root team, "Tlorentzvector."

[10] M. Franchini, K. H. Mankinen, G. Carratta, F. Scutti, A. Gorisek, E. Lytken et al., *Search for type-III seesaw heavy leptons in dilepton final states in pp collisions at $\sqrt{s}$ = 13 TeV with the ATLAS detector*, .

[11] M. Aaboud, , G. Aad, B. Abbott, D. C. Abbott, O. Abdinov et al., *Electron reconstruction and identification in the ATLAS experiment using the 2015 and 2016 LHC proton–proton collision data at $$\sqrt{s} = 13$$ $$\text{TeV}$$*, *The European Physical Journal C* **79** (aug, 2019) .