Introduction of Smart Classifiers at the LHCb to Improve Electron Detection

Daniel Manz

Faculty of Science and Engineering

Rijksuniversiteit Groningen

July 2022

university of groningen

faculty of science and engineering

Bachelor's Project for the BSc. Physics

July 2022

Student: D. Manz

First Examiner: Maarten van Veghel

Second Examiner: Lorenz Willmann

Abstract

Lepton Universality is a property predicted by the Standard Model. A departure from this property would imply a departure from this framework and the presence of New Physics. This work aims to assist the search for Lepton Universality Violation by increasing electron detection efficiency. This is achieved through the introduction of smart classifiers. Various machine learning tools are implemented and the AUC value for electron detection is increased from 0.9815 to 0.9731.

Contents

1 Introduction

The Standard Model of Particle Physics is the apex of predictive frameworks outlying the constituents of nature. This framework is our current paradigm of understanding. However, hints of New Physics suggest it is simply a provisional incarnation of a more global theory. The scientific community must challenge this paradigm; defects and shortcomings must be sought. This can inspire adjustments and alterations to the Standard Model and, subsequently, facilitate a better understanding of nature.

Lepton Universality is a property embedded in the Standard Model. It is a falsifiable, empirical claim and therefore a popular avenue in the search for Lepton Universality Violation. Evidence of such a violation would suggest a departure from the Standard Model and the existence of New Physics. This work aims to assist this search through considering the following research goal:

Introduction of Smart Classifiers at the LHCb to Improve Electron Detection.

Through the introduction of smart classifiers the quality of electron detection can be increased. This increases the amount of usable data and stimulates the search for Lepton Universality Violation. Smart classifiers denote classification models where Machine Learning approaches have been adopted.

Chapter 2 will consider, in more detail, the theoretical context of the thesis; the Standard Model and Lepton Universality will be more extensively discussed. Chapter 3 gives an overview of the experimental setup used. The Large Hadron Collider beauty experiment is introduced followed by a discussion of relevant sub-detector equipment and software components. Chapter 4 introduces the key features and ideas in building a Machine Learning model, followed by an application of these concepts in chapter 5. A discussion of limitations, and how these could influence future endeavours, comprises chapter 6.

2 Lepton Universality

2.1 Standard Model of Particle Physics

The Standard Model of Particle Physics (SM) is a predictive framework outlying the fundamental constituents and forces of nature. For decades it has been the most widely accepted, and successful, depiction of reality. It encapsulates all elementary particles and, omitting gravity, all fundamental forces. The SM is shown in Figure 1.

Despite much experimental success in support for the SM, it is not without its limitations. It can not account for the matter-antimatter asymmetry visible in the Universe, and it fails to suggest a Dark Matter candidate. These limitations suggest the SM is a lower-level incarnation of a more global theory. This inspires the search for New Physics (NP) - phenomena beyond the scope of the current SM.

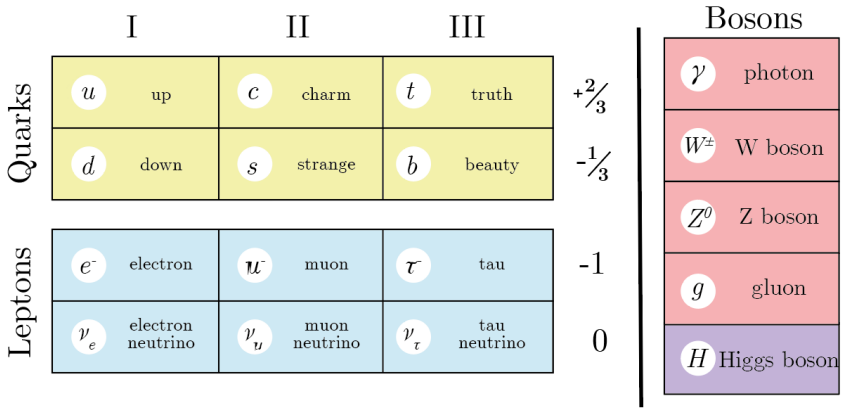

Figure 1: Standard Model of Particle Physics. Taken from [8].

2.2 Lepton Flavour Universality within the Standard Model

Withing the framework of the SM, electroweak gauge bosons Z and $W^{\pm}$ have identical couplings to all three lepton flavours: The electron (e^-), the muon (μ^-) and the tauon (τ^-)[1]. All leptons have identical electroweak interaction strengths. This peculiar property of of the SM is labelled Lepton Universality (LU). Any departure from this property would be a clear sign that the SM is insufficient and that NP particles are present. The only exception to this universaility is the differing lepton masses ($m_\tau > m_\mu > m_e$) due to the non-vanishing vacuum expectation value of the Higgs field, and the subsequent lepton-Higgs interaction.

The avenues probing LU are multifarious. The leptonic decays of gauge bosons, semileptonic decays involving quarks and purely leptonic decays such as $\tau^- \rightarrow e^- \bar{\nu_e} \nu_\tau$ have all been rifely studied. Semileptonic tree-level $b \rightarrow clv$ decays, and their corresponding branching ratios, are particularly auspicious in the search for LU violation:

$$R(D) = \frac{\mathcal{B}(B^0 \rightarrow D^+ \tau^- \bar{\nu_\tau})}{\mathcal{B}(B^0 \rightarrow D^+ \mu^- \bar{\nu_\mu})} \quad and \quad R(D*) = \frac{\mathcal{B}(B^0 \rightarrow D^{*+} \tau^- \bar{\nu_\tau})}{\mathcal{B}(B^0 \rightarrow D^{*+} \mu^- \bar{\nu_\mu})}. \tag{1}$$

These ratios express the relative frequency of the decays of B^0 mesons into different generations of leptons. A B meson is a short-lived meson comprised of one bottom anti-quark and either an up, down, or strange quark. B^0, the B meson responsible for this decay, has quark composition $d\bar{b}$. D/D^* denote the ground and excited state of the charmed D mesons respectively. Note that this branching ratio is not expected to be unity due to the influence of form factors. The LHCb[2,3], Belle[4,5], and BaBar[6] experiments have combined their results through the Heavy Flavour Averaging Group (HFLAV) and have found a combined 3.3σ tension with the SM predictions[7]. These experimental results are shown in Figure 2.

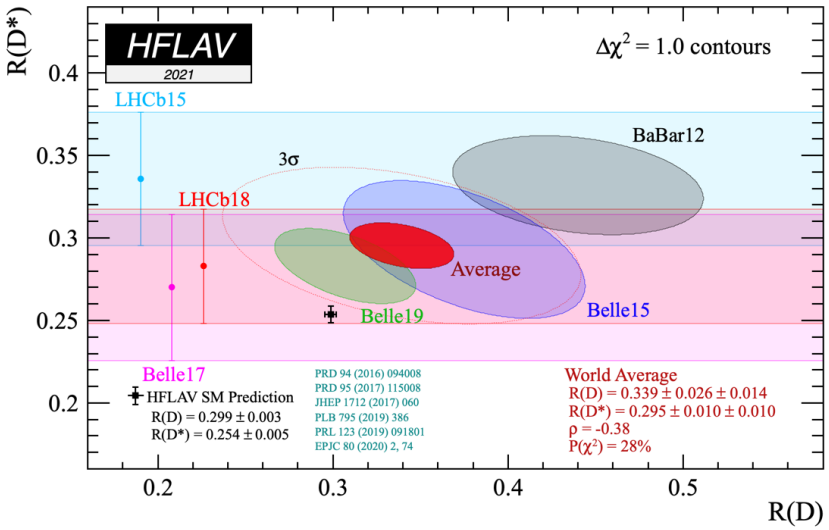

Figure 2: Branching ratio results $R(D)/R(D^*)$ for semileptonic $b \to clv$ decays. Given for the LHCb, BaBar and Belle experiments. Combined by the HFLAV group. Taken from [7].

Another widely investigated avenue in the search for LU violation is the semi-leptonic $b \to sl^+l^-$ decay. Particularly auspicious are the branching ratios $R(K)/R(K^*)$ which, where B denotes a meson, can be defined as follows:

$$R(K) = \frac{\mathcal{B}(B^+ \to K^+\mu^+\mu^-)}{\mathcal{B}(B^+ \to K^+e^+e^-)} \quad and \quad R(K^*) = \frac{\mathcal{B}(B^+ \to K^{*0}\mu^+\mu^-)}{\mathcal{B}(B^+ \to K^{*0}e^+e^-)} \tag{2}$$

The value of these branching ratios is predicted with $\mathcal{O}(1\%)$ precision. Due to the small masses of electrons and muons relative to that of b quarks, this ratio is expected to be unity. However, measured ratios have been found to be 2.1 - 2.5 σ below these SM predictions. A comparison of $R(K)/R(K^*)$ measurements for different experiments is shown in Figure 3. Due to the presence of electrons in both branching ratios, my thesis is particularly relevant for these decay channels. Using Machine Learning(ML) to increase electron detection efficiency can be used to estimate systematic uncertainties and is relevant for all decays where electrons are present in the final state.

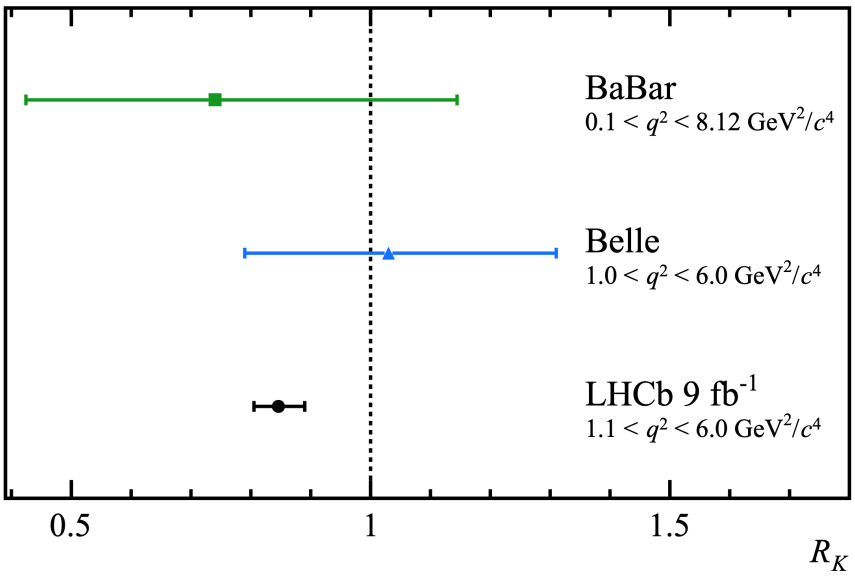

Figure 3: Branching ratio results for $R(K^*)$. Given for BaBar, Belle and LHCb. The quantity q^2 refers to the invariant mass of the dilepton pair. This value is integrated over to determine $R(K)/R(K^*)$. Taken from [9].

3 Large Hadron Collider Beauty Experiment

3.1 LHCb Upgrade

The Large Hadron Collider Beauty (LHCb) experiment[10] is a single-arm forward spectrometer operating in the pseudorapidity range $2 < \eta < 5$. The experiment seeks to detect CP violation and rare decays of beauty and charm hadrons. It does so, primarily, through proton-proton collisions. During Long Shutdown 2(LS2) of LHCb, from mid 2018 to the end of 2019, LHCb was subject to an upgrade. This facilitated a five-fold increase increase in instantaneous luminosity: from $\mathcal{L}_{int} = 4 \times 10^{32} cm^{-2} s^{-1}$ to $\mathcal{L}_{int} = 2 \times 10^{33} cm^{-2} s^{-1}$. The upgrade achieved a trigger-free 40MHz complete event readout; event selection is solely performed by a by a high-level software trigger farm discussed in section 3.3.1. Much of the detector system and readout electronics have been replaced. All subdetectors must be qualified in performance up to $\mathcal{L}_{int} = 2 \times 10^{33} cm^{-2} s^{-1}$.

These upgrades allow LHCb to run at a higher instantaneous luminosity with increased trigger efficiency for a wide range of decay channels. In the following section, the relevant tracking stations and calorimeters which comprise the LHCb are discussed. A schematic of these instruments can be seen in Figure 4.

3.1.1 Tracking Stations

VErtex LOcator (VELO)

This is the first detection instrument the beam enters. Located at the interaction points where protons collide, the VELO was upgraded to a hybrid pixel detector during LS2. The detector is comprised of 41 million $55\mu m \times 55\mu m$ pixels, read out by the CERN/Nikhef-designed ASIC VeloPix[11]. These pixel detector modules are mounted in two detector halves located on either side of the LHCb beams and are orientated perpendicular to the beam direction. These updated electronics allow data to be read out at 40MHz. Cooling is provided by evaporative CO_2 circulating in microchannel cooling substrates. The x and y coordiantes of the tracks are measured through the hybrid pixel detector. VELO contributes significantly to track reconstruction.

Upstream Tracker (UT)

This tracker is comprised of approximately 1000 silicon micro-strip sensors, forming four tracking planes[12]. It is placed upstream of the LHCb bending magnet, as is shown in Figure 4. The UT will contribute to the software trigger. Through the requirement that a particle is detected at the UT, in addition to detection at VELO and SciFi, one can reduce the number of possible track segment combinations. This, in turn, decreases the number of "ghost tracks". It was designed to replace the previous Tracker Turicensis to meet the upgraded performance requirements; it meets the $\mathcal{L}_{int} = 2 \times 10^{33} cm^{-2} s^{-1}$ criterion[10].

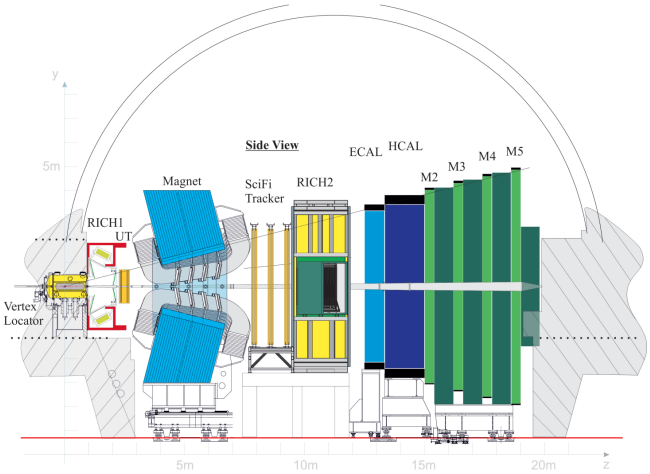

Figure 4: Schematic view of the LHCb detector in the y-z plane. Taken from [10].

Scintillating Fibre Detector (SciFi)

This instrument is a $340m^2$ high-resolution detector introduced in LS2. The detector consists of individual modules, each containing eight 2.4m fibre mats of active detector material. These fibre mats contain six layers of densely packed blue-emitting scinitillating fibres with a diameter of 250μm[13]. The emitted scintillation light is recorded with multi-channel silicon photo-multipliers and digitised by a custom ASIC. The SciFi tracker provides a momentum resolution for the charged particles better than 100μm in the bending direction of the LHCb spectrometer[10].

3.1.2 Calorimeters

Electromagnetic Calorimeter (ECAL)

ECAL is comprised of 6000 detector cells and is placed 12.5m from the interaction point[14]. "Shashlik" technology is adopted implying the use of a sampling scintillator and a lead structure. The module is constructed from alternating layers of 2mm thick lead, 120μm thick paper (TYVEK), and 4mm thick scintillator tiles. ECAL uses secondary electromagnetic showers for the reconstruction of both high energy ($> 1GeV$) photons and electrons. The decision to adopt Shashlik technology was inspired by its high energy resolution, fast time response and acceptable radiation resistance.

Hadronic Calorimeter (HCAL)

This sampling iron scintillator is placed 13.33m from the interaction point. HCAL has an active area front surface of $8.4\times6.8m^2$[15]. It is comprised of thin iron plates inter spaced with scintillating tiles. In contrast to the ECAL, the HCAL scinitillating tiles are placed parallel to the beam direction. HCAL uses this scintillation technology, in conjunction with the creation of secondary hadronic showers, for the energy reconstruction of hadrons.

It should be noted that the increased instantaneous luminosity $\mathcal{L}_{int} = 2 \times 10^{33} cm^{-2} s^{-1}$ did not demand substantial rebuilds of the calorimeter system. The preexisting granularity for both HCAL and ECAL was sufficient for this $\mathcal{L}_{int}$.

3.2 LHCb Software

3.2.1 Trigger System

The increase to $\mathcal{L}_{int} = 2 \times 10^{33} cm^{-2} s^{-1}$ and an event rate of 40MHz also demands a change in trigger system requirements. These changes were designed and implemented during LS2. Firstly, a transition was made to a trigger-less system - the Level 0 (L0) hardware trigger was discarded. Such a system will allow the full inelastic collision rate of 30 MHz to be processed by the software trigger.

The LHCb trigger system is now exclusively software based and implemented in a CPU farm. The software trigger is still divided into High Level Trigger 1 (HLT1) and High Level Trigger 2 (HLT2). HLT1 executes a partial reconstruction and event selection. HLT2 performs full detector reconstruction and selection. The increase in electron detection efficiency and scope of this thesis is relevant for HLT2. This data-flow is shown in figure 5.

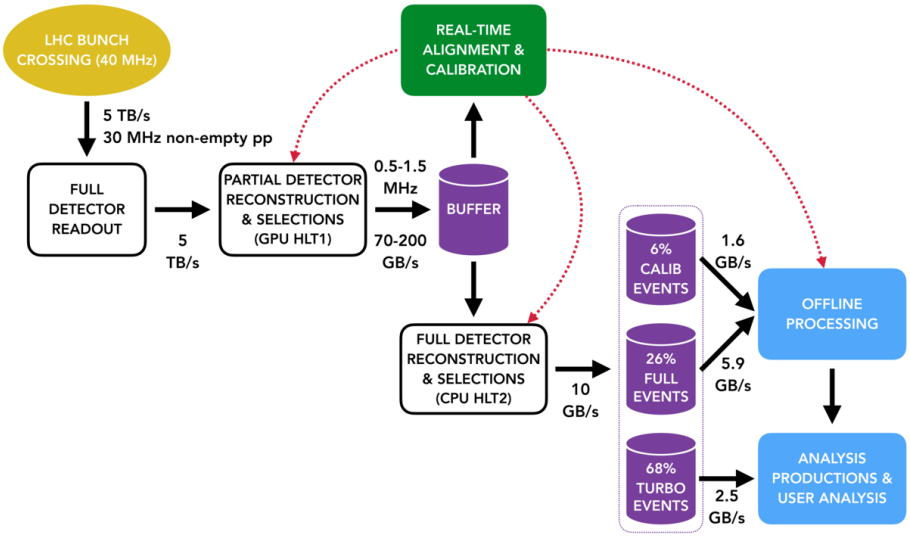

Figure 5: Data-flow for the LHCb upgrade. Taken from [17].

3.2.2 Data Simulation

This section will discuss how the simulated data used in this work is obtained. The simulation is governed by Gauss, the LHCb simulation framework. The PYTHIA application[18] controls the generation of pp collisions through the employment of Monte Carlo methods. The generated particles and their subsequent traversion through the experimental apparatus is governed by the GEANT4 toolkit[19]. For run 3, all LHCb-independent components from the simulation software have been placed in a separate project: GAUSSINO[20]. GAUDI[21], the LHCb core software framework, has been re-engineered to a multi-threading mode to facilitate the benefits of inter-event parallelism. A view of the discussed dependencies can be seen in Figure 6.

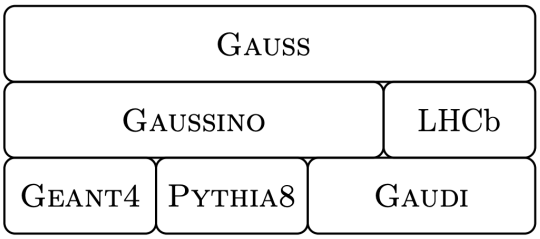

Figure 6: Software dependencies for the LHCb upgraded simulation framework. Taken from [26].

The final step in the simulated data flow is the translation of simulated hits to a digital detector output; this process is governed by the BOOLE application. BOOLE mimics the specific sub-detector technologies and electronic response. The digitised data is then fed to the HLT software trigger. At this point, the simulated data is following the same data-chain as the real data.

The process of data simulation is key to the interpretation of physical measurements and the development of ML models to investigate decays. Simulation grants a data set which contains primarily the requested decays. In addition, real data suffers from contamination and biases. In the case of this thesis, simulated data will be used to develop ML models for electron identification.

4 Electron Identification

This chapter will discuss the methods employed in the improvement of electron identification. ML algorithms will be adopted for this task due to their speed, efficiency and general performance.

In the context of this thesis, an ML algorithm should be chosen which has the greatest separation power between an electron signal and a background signal. This background signal is comprised of pions. The ML algorithm is fed a set of user-defined variables which compose a feature vector. This feature vector is mapped to a scalar variable known as the "test statistic". The task can be summarised as defining the best possible feature vector and applying the most suitable ML model. This will facilitate a test statistic which has the highest differentiating power between an electron and a pion.

4.1 Baseline

This work aims to improve the current electron identification model. This baseline test statistic is a Delta-Log-Likelihood (DLL) ratio. The calculation of these DLLs are based on calibration for histograms, for which the number of bins is $\prod_i^{n_{features}}(n_{bins}(i))$. This blows up as features are added; it is therefore wise to reduce the number of features when calculating a DLL. This baseline model is therefore restricted by dimensionality; this work will explore to what extent classifier performance can increase when these dimensionality constraints are lifted. An overview of the baseline test statistic is displayed in Figure 7. In addition, the resultant receiver operating characteristic curve and its accompanying area under curve value are shown in Figure 8. The exact meaning of this graph is elucidated in sub-chapter 4.4.

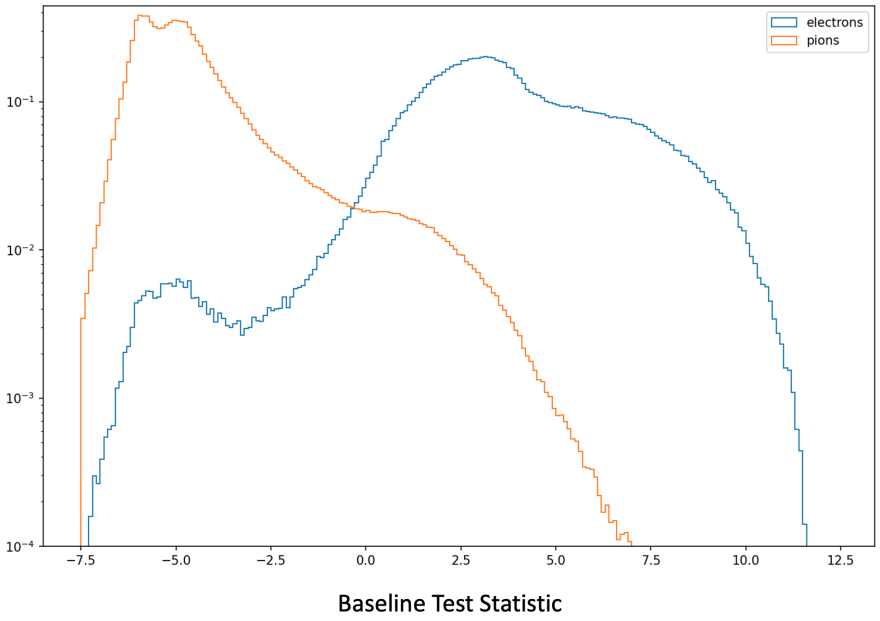

Figure 7: The baseline test statistic, displayed for both electrons and pions.

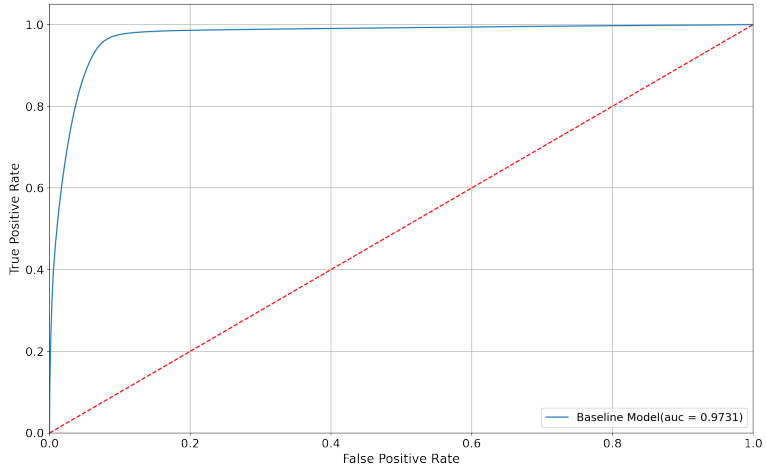

Figure 8: ROC plot for the baseline test-statistic with accompanying AUC value.

4.2 Signature Properties

The following section will discuss features with a high discriminating ability between electrons and pions and why they were selected. This discussion is crucial for defining the best possible feature vector.

ElectronShowerEoP is derived from expected energy deposit at the ECAL. This expectation is based on electron nature, track information and ECAL geometry. Subsequently, the ECAL cells are selected with this expectation energy deposit. ElectronShowerEoP is the sum of the energies of these cells over the track momentum; the expectation value for an electron is 1 as it should lose all its energy at the ECAL. This peak is clearly visible in Figure 9.

ElectronShowerDLL "squeezes" more information out of the ECAL cells through the construction of a likelihood ratio. The variable is a summation, over the selected cells, of the DLL of cell energy given a certain expected energy. This is a transformation of the ElectronShowerEoP variable to a DLL per cell. This variable can also be seen in Figure 9.

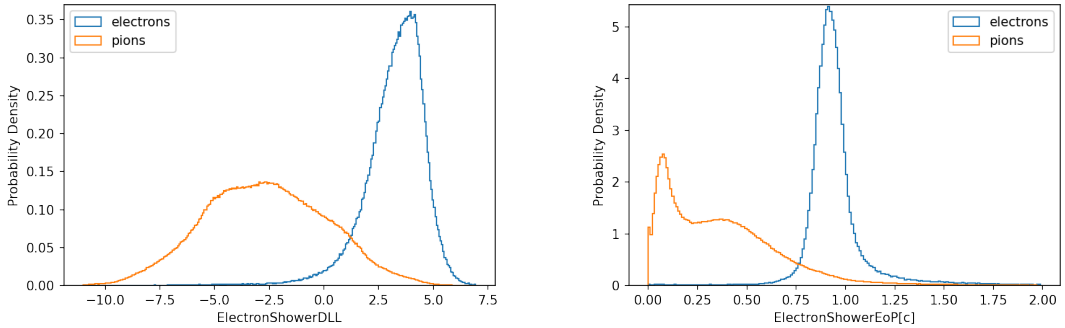

Figure 9: Distributions for the variables "ElectronShowerDLL" and "ElectronShowerEoP", plotted for both electrons and pions. Both properties are derived from electron showers at the ECAL.

BremHypoDeltaX is a test statistic comparing the first-state like and last-state like test statistic. These statistics are positive and negative respectively. Electrons tend to be first-state like and hadrons random, as hadrons do not experience real bremsstrahlung. This can be seen in Figure 10.

BremHypoMatch is a χ^2 value of the cluster determined at the ECAL. It compares the track

extrapolation at the final state before the magnet to the cluster position.

BremHypoEnergy is the corresponding energy of the cluster described by BremHypoMatch. These properties are both displayed in Figure 10.

BremTrackBasedEnergy is derived from the ECAL. The ECAL cells match the track extrapolation from before the magnet to the ECAL. It is not ECAL cluster based.

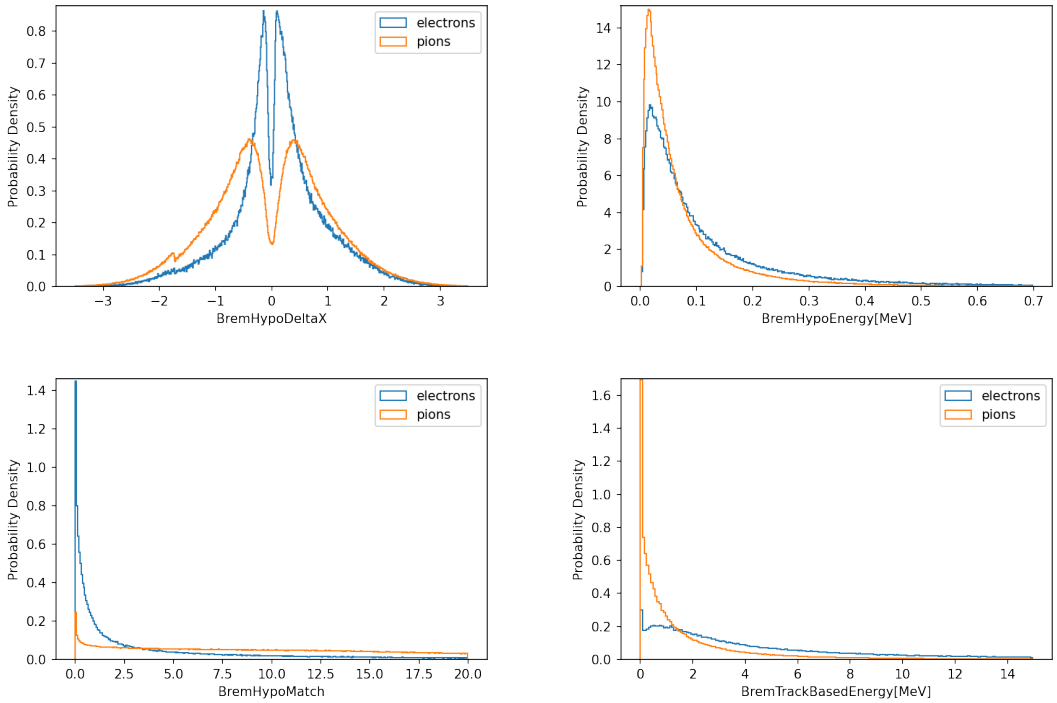

Figure 10: Distributions for the variables "BremHypoDeltaX", "BremHypoEnergy", "BremHypoMatch" and "BremTrackBasedEnergy". Plotted for both electrons and pions.

ElectronMatch is the χ^2 value of the cluster at the ECAL. It compares the track extrapolation to the ECAL and the observed cluster position.

ElectronEnergy is the corresponding energy of the respective cluster. These variables can be seen in figure 11.

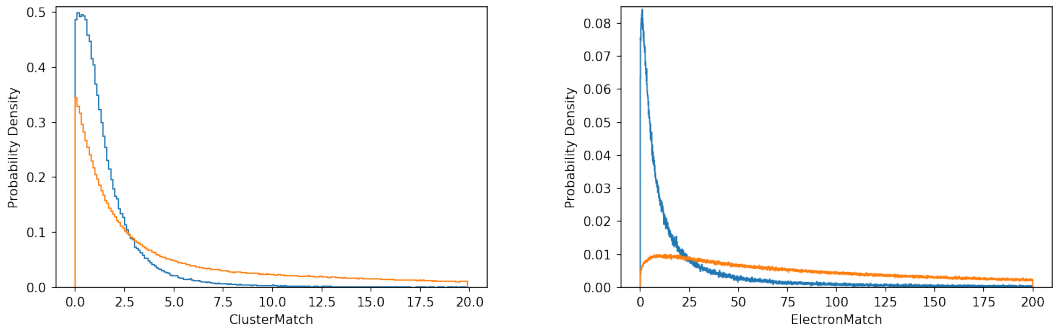

Figure 11: Distributions for "ClusterMatch" and "ElectronMatch". Both variables derive from cluster properties at ECAL. Plotted for both electrons and pions.

HcalEoP is the energy measured at the HCAL over the momentum of the track. The energy is determined using the cells in the HCAL that cross the track extrapolation. This should be low for electrons, which are expected to lose their energy in the ECAL. This pattern can be seen in Figure 12.

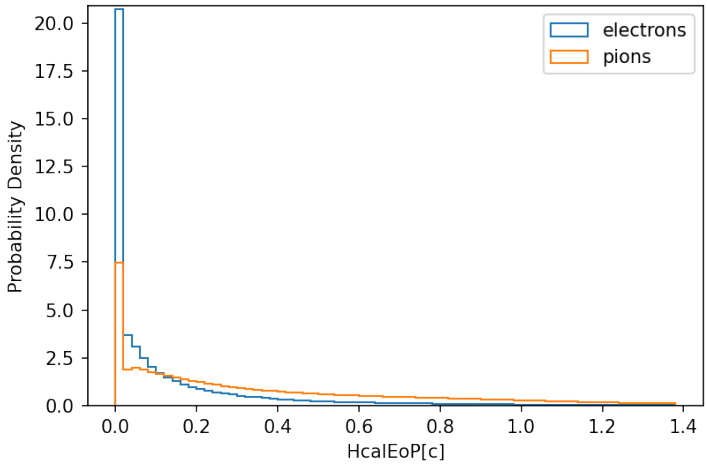

Figure 12: Distribution for the variable "HcalEoP", plotted for both the electrons and the background pions.

4.3 Classification Models

Decision Tree

This algorithm employs recursive binary splitting to categorise an event. In the case of numeric observation, this binary splitting will be based on numeric range conditions. A split criterion is required to increase the speed and quality of the classifications. This is done through defining the Gini index[22],

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}), \tag{3}$$

a measure of total variance across the K classes. Note that p_{mk} represents the proportion of training observations in the mth decision region that are from the kth class. Therefore, G is a measure of purity; an optimum binary cut will minimise G. Alternatives to G include entropy and log-loss. The default split criterion in the scikit-learn Python package, which was used to implement the algorithm, is G.

Random Forest

Random Forest[23] is an extension of the aforementioned decision tree designed to reduce the variance of the model. Variance refers to error due to a model's sensitivity to small fluctuations in the training set. Bootstrap aggregation, referred to as "bagging", significantly reduces this quantity. Bootstrapping involves random sampling of the data and growing subsequent decision trees based on this sub-sample. When unseen data is presented, it is passed down all of the individual decision trees and an aggregate is calculated. The final decision is based on this aggregate. In addition, the individual decision trees take a random sub-sample of the total available features; this "decorrelates" the data. This algorithm is also implemented using the scikit-learn Python package.

ADABoost

AdaBoost[24] utilises "weak learners": simple, two-option decision trees also referred to as "stumps". The algorithm begins with building an initial decision stump that performs best, under a specified criterion, on the training set. Note that this initial training set contains equally weighted samples. AdaBoost is adaptive in the sense that when building subsequent stumps it ascribes weights based on the previous performance. Learners receive a larger weight if they performed performed well. Samples receive a new weight based on the performance of the learner; misclassified samples recieve an increased weight and and vice versa. A re-sampling is then performed so that large-weight samples have a higher chance of appearing. This shifts the focus on to the previously misclassified

samples. The final classification decision is now based on the weighted aggregate of the decision stumps. This algorithm is also implemented using the scikit-learn Python package.

Gradient Boosting Classifier (GBC)

GBC[25] further advances the AdaBoost algorithm. The algorithm establishes an initial, single leaf. The residuals are then calculated for the training set samples and a decision tree is built to predict their values. One can now predict a weight for the decision tree using the residual. This, however, can lead to a high variance. As a counteraction meausure, a "learning rate" is introduced; this scales the contribution of the new tree. It aims to maximise the number of correctly classified samples and increase the certainty that the sample belongs to a predicted class. This is achieved through the minimisation of a loss function using gradient descent. Scikit-learn was used to implement this algorithm.

eXtreme Gradient Boost (XGBoost)

This model is an implementation of GBC designed to optimise speed and performance. XGBoost minimises an objective function that combines a loss function and a penalty term for complexity. It takes a variety of measures to push GBC to the limit. It implements parallelisation of tree construction; all CPU cores are used for training. Distributed computing is used for training very large models. Finally, out-of-core computing is used when encountering data sets too large to fit into memory. XGBoost was implemented using the open source XGBoost Python package.

4.4 Performance Metric

The task of quantifying performance is non-trivial; performance metrics are manifold, with no general consensus among practitioners regarding which to use. It is, therefore, important to outline the choices made for this thesis. Classifier performance will be presented through Receiver Operating Charcateristic (ROC) curve plots. True Positive Rate (TPR) lies on the vertical axis, and False Positive Rate (FPR) on the horizontal. The Area Under the ROC Curve (AUC) is the performance metric chosen for this work. A perfect classifier will have an AUC value of 1 whilst a completely random classifier will have an AUC value of 0.5.

The ROC curve and accompanying AUC value were chosen to gauge the performance of classification models for various reasons. Firstly, it is directly applicable to all classification models under consideration. Secondly, the performance of models can be easily visualised and compared. Finally, the trade-off between precision and recall is easily investigated; one can clearly see how the performance of a model changes when the threshold for electron classification is tweaked. This trade-off is useful in practice. For example, if one is searching for a rare electron signal the requirements for a positive identification can be restricted as to not include a background signal.

It should be noted that, although AUC will be the primary consideration in model selection, training time will still be considered a factor.

5 Results

The concepts and models outlined in the previous chapter can now be applied. The various stages of ML development will be discussed: model selection, feature selection and hyper-parameter tuning. The impact on the performance of the model will guide these decisions, and the end result will be compared with the baseline model.

5.1 Model Selection

Firstly, an ML algorithm must be selected. Five models are considered: Decision Tree, Random Forest, AdaBoost, GBC and XGBoost. Tree ensemble methods were selected due to superior AUC values and their flexibility. They also possess logistical advantages; the data requires no pre-processing, normalisation or scaling.

To determine which model is best suited for electron identification several AUC tests are performed. These tests were performed on a randomly selected sub-sample of length 300,000 to provide realistic training times. It was unfeasible to perform several tests on the full data, of length 19,077,334, with the computational resources and time available. A typical ROCAUC comparison can be seen in Figure 13.

The GBC and XGBoost outperform these other models. Their supremacy withholds under the scrutiny of cross-validation. Cross-validation denotes permuting how the data is partitioned into testing and training sets. This technique protects the model from overfitting. Overfitting is the error encountered when a ML model corresponds too closely to a particular training set. A model which suffers from overfitting will fail to generalise to unseen data and will not sustain its performance. The model comparison using cross-validation can be seen in Table 1.

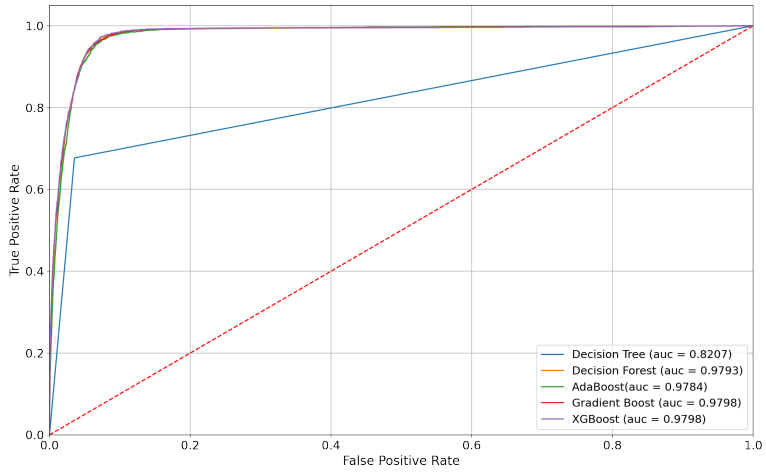

Figure 13: Typical ROC curves the five Machine Learning Classifiers under consideration. The AUC values are displayed for convenience.

Classifcation Model	Mean AUC
Decision Tree	0.8162
Decision Forest	0.9759
AdaBoost	0.9753
Gradient Boost	0.9773
XGBoost	0.9774

Table 1: Mean AUC performance values using seven-fold cross-validation. Calculated for all machine learning classifers under consideration.

Cross-validation can also be used to determine the mean training time of an algorithm. The mean training time for each model is shown in Table 2. A significant reduction in training time can be seen when XGBoost is implemented. Due to this high efficiency, and a comparitively high AUC value, the choice was made to continue with the XGBoost algorithm.

Classifcation Model	Mean Training Time (s)
Decision Tree	10.1
Decision Forest	169.5
AdaBoost	46.8
Gradient Boost	212.4
XGBoost	29.0

Table 2: Mean training time for the machine learning classifiers under consideration. These values were calculated using a seven-fold cross-validation.

5.2 Feature Selection

Feature selection is the process of selecting a sub-set of properties for the feature vector. This procedure has numerous advantages. Firstly, feature reduction facilitates shorter training times. Feature reduction also increases the simplicity and, subsequently, the explainability of a model. Finally, feature selection is important in protecting a model from the "curse of dimensionality". This is a phenomenon associated with the exponential increase in volume associated with an extra dimension in Euclidean space. If the dimensionality of a model is too high model performance deteriorates and overfitting can occur.

One can begin with analysing relative feature importance for a trial model. This importance is computed as the mean and standard deviation of the accumulation of the impurity decrease within each tree. The impurity criterion used is the Gini index, discussed in section 4.3. The results of the comparison can be seen in Figure 14. In essence, the most important features have the highest differentiating power between electrons and pions. The feature ElectronShowerDLL, which boasts the highest importance, exhibits clear variation in behaviour for electrons and pions; this is visible in Figure 9.

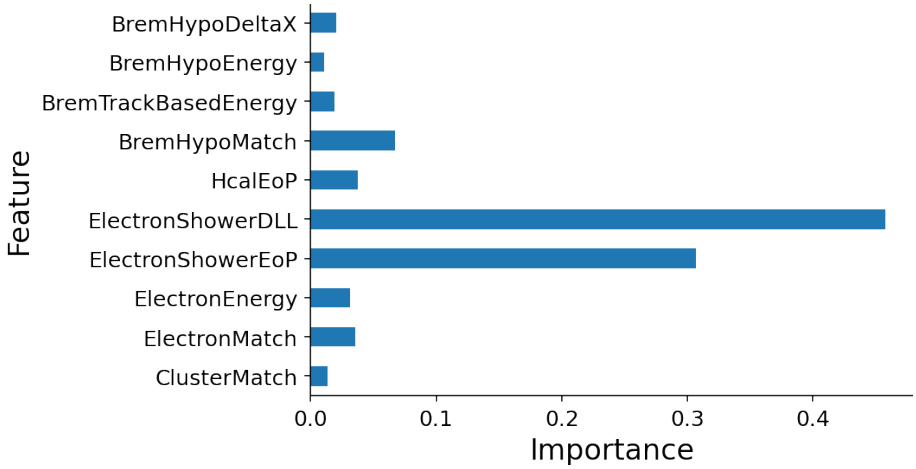

Figure 14: Feature importance comparison for the ten features under consideration.

The effects of feature selection, and the optimum dimensionality, can be gauged through iteratively increasing dimensionality in the order of descending feature importance. Throughout this process, the influence on model performance can be analysed. A typical ROC curve for this procedure can be seen in Figure 15. 300,000 randomly selected data points were used for this performance comparison. It can be seen that there is an increase in model performance as dimensionality increases.

Number of Features	Mean AUC
1	0.9545
4	0.9727
7	0.9757
10	0.9765

Table 3: Mean AUC values for a varying number of features. These values were calculated using a seven-fold cross-validation.

Number of Features	Mean Training Time (s)
1	27.3
4	36.0
7	37.6
10	38.3

Table 4: Mean training time for a varying number of features. These values were calculated using a seven-fold cross-validation.

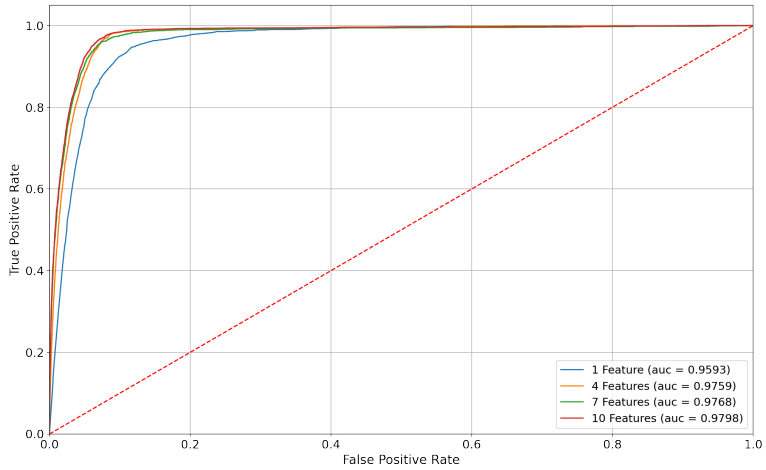

Figure 15: Typical ROC curves for an increasing number of features. Shown for one, four, seven and ten features. Accompanying AUC values are displayed.

This result must, again, be validated through cross-validation. The results of a seven-fold cross-validation can be viewed in Table 3. Again, AUC values increase as dimensionality increases. Cross-validation was also used to determine the mean training time for training the XGBoost model. These times are presented in Table 4. A dimensionality of ten exhibits superior performance for both the ROC curve in Figure 15 and the cross-validated mean AUC. In addition, the increase in training time is not considered problematic; there was a mere 2.3 second increase for an increase in dimensionality from four to ten. Therefore, it was decided to utilise all available features and continue with a feature vector of length ten. It is believed that the "curse of dimensionality" is avoided due to the large quantities of data available; this "curse" is more pertinent when the amount of data is limited. Despite a higher dimensionality actions will be taken, at a later stage, to ensure the explainability of the model is preserved.

5.3 Hyper-Parameter Tuning

The model architecture itself can be fine tuned to optimise performance. The internal parameters which dictate this architecture are known as hyper-parameters. The alteration of individual hyper-parameters, and the subsequent fluctuations in model performance, can be explored through validation curves. These curves allow one to extract the useful hyper-parameters - those that cause a substantial change in AUC values.

One can also extract the optimum value of a hyper-parameter from these curves. If the values for the training score/cross-validation score are both low the model is under-fitting; it is too primitive. If the cross-vaidation curve is lagging behind the training curve the model is over-fitting. This means there is unnecessary model complexity damaging the model performance. One should seek hyper-parameter values which give a high AUC value and a minimised distance between the training curve/cross-validation curve. The validation curves for these useful hyper-parameters are given in Figures 16 and 17.

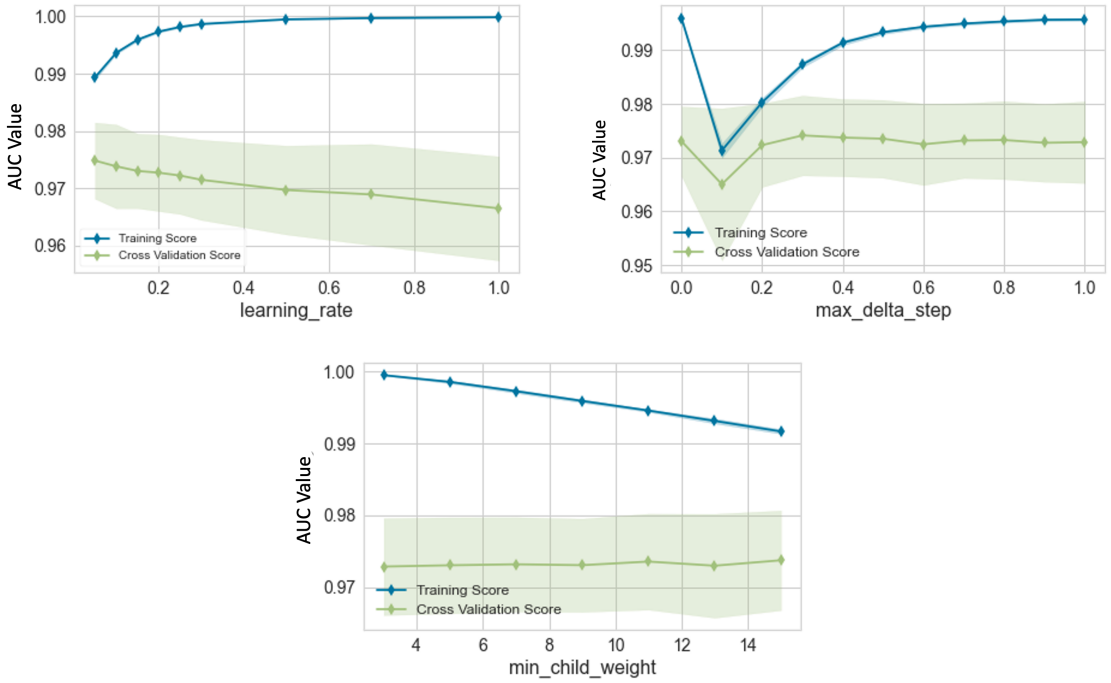

Figure 16: Validation curves for the XGBoost hyper-parameters "learning_rate", "max_delta_step" and "min_child_weight".

Figure 16 depicts useful hyper-parameters which relate to the weighting of new trees. The variable "learning_rate" is a weighting factor applied to new trees to scale down their contribution and protect the model from over-fitting; one can see the presence of over-fitting when this quantity becomes too high. It is a measure of relative regularisation in the sense it is a weight multiplied by a constant factor. In contrast, "max_delta_step" is an additional measure of absolute regularisation. It caps the weight for new trees before a learning rate correction. One can see that when this value is too low the model becomes rigid, under-fitting occurs, and model performance deteriorates.

Finally, hyper-parameter "min_child_weight" is the minimum sum of instance weight needed in a child. If the tree partition step results in a leaf node with the sum of instance weight less than "min_child_weight" the building process will terminate. This protects the model from adding unnecessary complexity.

Figure 17 shows the remaining 2 useful hyper-parameters which will be used for tuning. The variable "max_depth" restricts the depth of each new tree; the amount of decision nodes the tree is comprised off. Restricting this will prevent vacuous complexity and undesired over-fitting. When "max_depth" is too high it can be seen that the difference netween the training curve/cross-validation curve increases; this is a clear indication of over-fitting. Finally, the parameter "sub-

sample" dictates the subsample ratio of the training instances. For example, if the "subsample" value is 0.5 XGBoost will randomly sample half of the training data prior to growing trees. This, again, is a design element to prevent over-fitting.

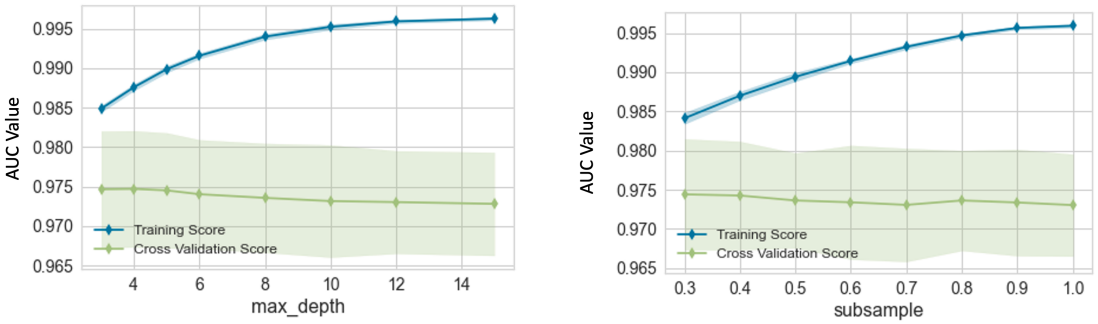

Figure 17: Validation curves for the XGBoost hyper-parameters "max_depth" and "subsample".

Once these useful hyper-parameters have been identified one can define a hyper-parameter space to explore. The choices of hyper-parameter values to explore, which are motivated by promising regions of the aforementioned validation curves, are presented in Table 5. One must now locate the values, within the hyper-parameter space, which give the best model performance. This task will be approached using randomised search. Random permutations of values within the hyper-parameter space are selected, a candidate model is built and an AUC value is calculated. Ten iterations of this procedure are performed, each with a different permutation of the hyper-parameter space. The parameters which facilitate the highest AUC value are then returned.

These results are also shown in Table 5. Note that these results, to a high degree, concur with expected values suggested by the validation curves. For example, the randomised search returns a "max_depth" value of five; this value facilitates high values for the validation curves but impedes the training curve advancing too far ahead of the cross-validation curve. This can be visualised in Figure 17.

A five-fold cross-validation is executed for each iteration; an addtional cross-validation is unnecessary. In addition, the alteration of hyper-parameters does not cause significant fluctuations in run time. Therefore, this is not being considered as a factor in hyper-parameter selection.

Hyper-Parameter	Search Space	Final Choice
learning_rate	0.05, 0.1, 0.2, 0.25, 0.3	0.1
max_delta_step	0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7	0.5
min_child_weight	3, 5, 7	3
max_depth	2, 4, 5, 8	5
subsample	0.2,0.4,0.6,0.8,1.0	1.0

Table 5: Definition of the hyper-parameter space to be explored. In addition, the final choice of each respective hyper-parameter is displayed.

5.4 Final Model

5.4.1 Baseline Comparison

The final XGBoost model has now been built. An ensemble of variables is created through feature selection and the internal model architecture has been tweaked through hyper-parameter tuning. It is now necessary to compare this model with the initial baseline test statistic. A typical ROCAUC comparison, using 300,000 data points for the XGBoost model, is shown in Figure 18. The AUC value has increased from the baseline of 0.9731 to 0.9815.

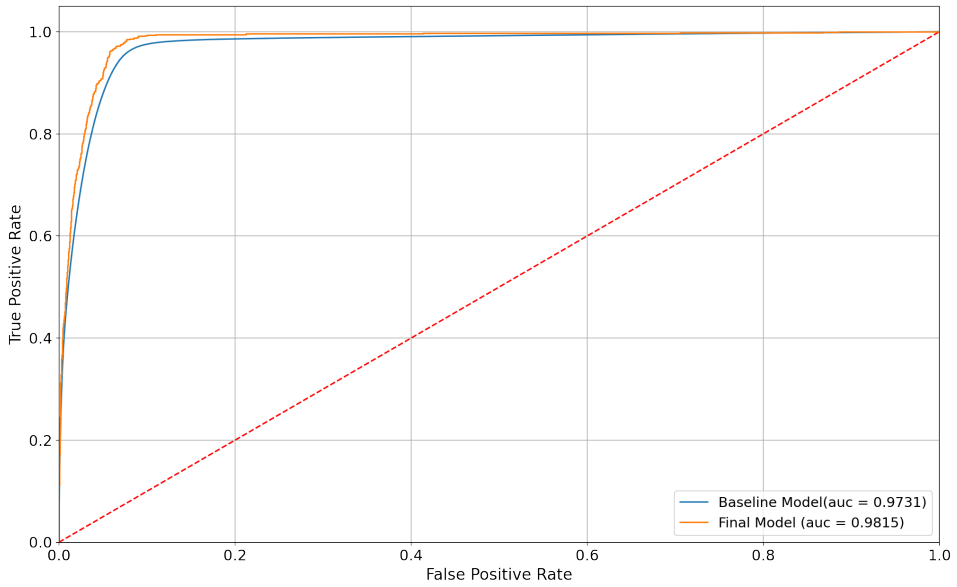

Figure 18: Typical ROCAUC comparison between the baseline model and the final XGBoost model.

5.4.2 Explainability

Despite the high dimensionality of the final model, which can skew understanding, it is believed to be important to preserve the property of explainability. This is understading *why* classification decisions were made. Interesting and valuable insights are provided. For example, one can gauge the expected impact and biases of a model. This can be crucial for an institution to build trust and confidence in the application of an ML model.

The SHapley Additive exPlanations (SHAP) Python package uses a game theoretic approach to increase the explainability of models. One can see, for example, how the properties of a simulated particle can push and pull a classification decision in opposing directions. This can be seen in the case of an electron decision and pion decision in Figure 19. These are known as SHAP "force plots". Note that the values on the horizontal scale denote the log-odds that the particle under consideration is an electron. It should also be known that "base value" is the mean log-odds. It is hoped that these images, to some extent, demystify the inner logic of the model.

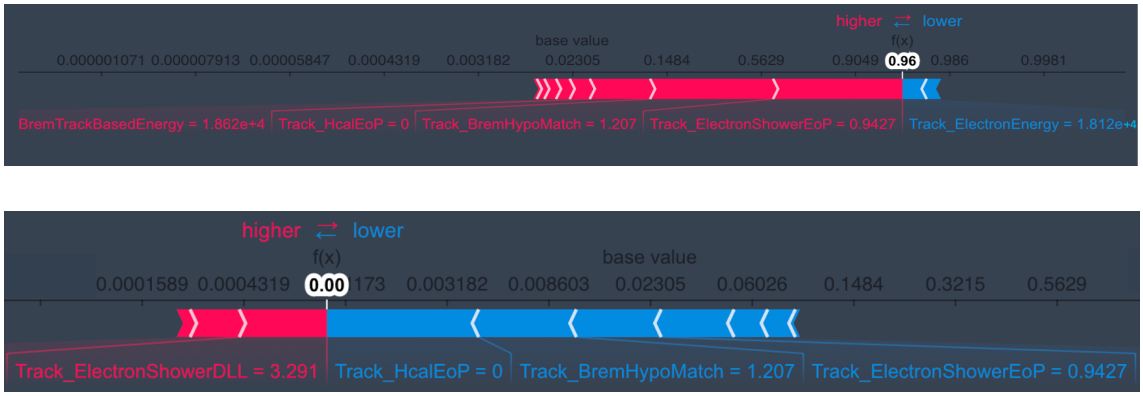

Figure 19: SHAP Force Plots for an electron (above) and a pion (below). The horizontal scale is the log-odds that the particle under consideration is an electron.

6 Limitations and Further Research

This section presents a critical discussion of the research; an analysis of limitations and how these can limit conclusions. Various issues will be discussed, and measures to mitigate these issues will be suggested. It is intended that future research can benefit from this discussion.

Firstly, the ML models were trained and tested using a sub-set of data; this measure was taken to reduce computational expense. This sub-set was of length 300,000. Methods were implemented to avoid any resulting biases: the sub-set was repeatedly randomly re-sampled and the results were consistently cross-validated. Despite the messures, the influence of only using a fragment of the simulated data cannot be neglected. In addition, it makes a direct comparison with the baseline test statistic problematic. The baseline model used all 19,077,334 data points to train and build the test statistic. This sub-sampling could be avoided in the future by using a centralised super computer to increase computational abilities.

Another issue is encountered when considering the translation of the results presented in this work to real data. The presented results are based on the simulation of data; this matter is discussed in section 3.3.2. There is no guarantee that the model performance will translate to data comprising of *real* events. However, this limitation is mitigated; simulated data follows the same data-chain as the real data. This similarity is beneficial as the simulated data suffers the same deficiencies as the real data. In addition, one can be absolutely certain of a particle's nature when working with simulated data. This property is highly suited to the development of ML models.

Finally, it must be acknowledged that the results are performance-metric dependent. This work adopted an ROCAUC approach to gauge classification performance. A valid alternative would be the F1-score which combines precision and recall into one metric. Despite the motivating factors for adopting the ROCAUC curve, discussed in section 4.4, the F1-score also possesses advantages. F1-score can be plotted *directly* against the decision threshold; this is useful as an optimum threshold can be determined. F1-score is highly suited when one cares more about the positive class; this is the case for electron identification. In addition, the utilisation of ROCAUC analysis can be hazardous when the target variables are imbalanced. This is because false positive rate, which comprise the horizontal axis for an ROC curve, is pulled down due to the large number of true negatives. This effect does not influence the quality of the F1-score. It would be recommended, for future endeavours, to conflate the conclusions of various, valid performance-metrics.

7 Conclusion

The Standard Model of Particle Physics has been, up to the present, the most successful and accurate description of nature. However, it is not without its limitations. It cannot account for the matter-antimatter asymmetry which permeates the Universe and it fails to propose a Dark Matter candidate. These failures, among others, are compelling hints of New Physics. These shortcomings must be taken seriously; the Standard Model must be rigorously trialled and tested.

Perhaps the most propitious avenue for probing the Standard Model has been the search for Lepton Universality violation. This is frequently investigated through the branching ratios for various lepton flavours. Electron identification is of crucial importance in the investigation of these branching ratios. However, the current baseline electron identification models are limited by their dimensionality. The work presented here aims to improve electron identification efficiency through the introduction of smart classifiers and, subsequently, to assist the search for Lepton Universality violation.

The results indeed indicate that improvement is possible. The machine learning model proposed increases the AUC value from 0.9731 to 0.9815. Albeit minor, this improvement is promising.

One must acknowledge some limitations when considering these results. There is no guarantee that this performance will directly translate to real data; the Machine Learning model was trained on simulated data. In addition, training the model on the entirety of the data is prohibitively computationally expensive. Despite these limitations, the increase in AUC value can be regarded as a success.

It is hoped that this work can increase the amount of usable data and stimulate the search for Lepton Universality violation. In doing so, this work hopes to assist the endeavour to find New Physics.

References

[1] K. Müller and. "Test of Lepton Flavour Universality at LHCb". In: *Journal of Physics: Conference Series* 1271(Dec. 2018), p. 012009. DOI: 10.1088/1742-6596/1271/1/012009. URL: https://doi.org/10.1088/1742-6596/1271/1/012009.

[2] R. Aaij et al. "Search for lepton-universality violation in $B^+ \to K^+ l^+ l^-$ decays." In: *Phys. Rev. Lett.* 122 (May 2019), p. 191801. DOI: 10.1103/PhysRevLett.122.191801. URL: https://doi.org/10.1103%2Fphysrevlett.122.191801.

[3] The LHCb collaboration et al. "Test of lepton universality with $B^0 \to K^{*0} l^+ l^-$ decays." In: *Journal of High Energy Physics* 2018.10 (Aug. 2017). ISSN: 1126-6708. DOI: 0.1007/JHEP08(2017)055. URL: https://doi.org/10.1007/JHEP08(2017)055.

[4] The Belle collaboration et al. "Test of lepton flavor universality and search for lepton flavor violation in $B \to Kll$ decays." In: *Journal of High Energy Physics* 2021.3 (March 2021). ISSN: 1029-8479. DOI: 10.1007/JHEP03(2021)105. URL: https://doi.org/10.1007/JHEP03(2021)105.

[5] S. Wehle et al. " Test of lepton-flavor universality in $B \to K^* l^+ l^-$ decays at Belle". In: *Phys. Rev. Lett.* 126 (April 2021), p.161801, DOI: 10.1103/PhysRevLett.126.161801. URL: https://link.aps.org/doi/10.1103/PhysRevLett.126.161801.

[6] J. P. Lees et al. "Measurement of branching fractions and rare asymmetries in the rare decays" $B \to K^{(*)} l^+ l^-$. In: *Phys. Rev. D* 86 (Aug. 2012), p. 032012. DOI: 10.1103/PhysRevD.86.032012. URL: https://link.aps.org/doi/10.1103/PhysRevD.86.032012.

[7] Y. Amhis et al. *Averages of b-hadron, c-hadron and τ-lepton properties as of 2021.* 2022. arXiv: 2206.07501 hep-ex.

[8] S. Bouma. "Analysis on the reconstruction of e^+e^- in $\Lambda_b^0 \to \Lambda^0 e^+ e^-$ using Run 2 and Run 3 simulations of the LHCb". BSc Thesis. Rijksuniversiteit Groningen, 2022.

[9] LHCb collaboration et al. *Test of lepton universality in beauty-quark decays.* 2021. arXiv: 2103.11769 hep-ex.

[10] LHCb Collaboration et al. *LHCb Tracker Upgrade Technical Design Report.* CERN-LHCC-2014-001, LHCB-TDR-015 (Feb. 2014). URL: https://cds.cern.ch/record/1647400.

[11] Á. D. Suárez. "The LHCb VELO upgrade". In: *Nuclear Instruments and Methods in Physics Research - section A.* 824 (July 2016). ISSN: 0168-9002. DOI: https://doi.org/10.1016/j.nima.2015.09.026. URL: https://www.sciencedirect.com/science/article/pii/S0168900215010815.

[12] M. S. Rudolph. "The LHCb Upstream Tracker Upgrade". In: *PoS Vertex 2019.* 013 (February 2020), p. 9. DOI: 10.22323/1.373.0013. URL: https://cds.cern.ch/record/2747954

[13] T. Kirn. "SciFi – A large scintillating fibre tracker for LHCb". In: *Nuclear Instruments and Methods in Physics Research - section A.* 845 (Feb. 2017). ISSN: 0168-9002. DOI: https://doi.org/10.1016/j.nima.2016.06.057. URL: https://www.sciencedirect.com/science/article/pii/S016890021630599X.

[14] S. Barsuk. "The Shashlik Electro-Magnetic Calorimeter for the LHCb Experiment." In: *2004 11th International Conference On Calorimetry In High Energy Physics.* 2004, pp. 61-67.

[15] Y. Guz. "The LHCb hadron calorimeter". In: *Journal of Physics: Conference Series* 160 (May 2009). DOI: 10.1088/1742-6596/160/1/012054.

[16] LHCb Collaboration et al. *LHCb Trigger and Online Upgrade Technical Design Report.* CERN-LHCC-2014-016, LHCB-TDR-016" (May. 2014). URL: https://cds.cern.ch/record/1701361.

[17] LHCb Collaboration et al. *RTA and DPA dataflow diagrams for Run 1, Run 2, and the upgraded LHCb detector.* (Sep. 2020). URL: https://cds.cern.ch/record/2730181.

[18] T. Sjostrand et al. "PYTHIA 6.4 physics and manual." In: *Journal of High Energy Physics* 2006.5 (May 2006). DOI: 10.1088/1126-6708/2006/05/026. URL: https://doi.org/10.1088/1126-6708/2006/05/026.

[19] S. Agostinelli et al. "Geant4—a simulation toolkit". In: *Nuclear Instruments and Methods in Physics Research - section A*. 506 (July 2003). DOI: 10.1088/1126-6708/2006/05/026. URL: https://doi.org/10.1088/1126-6708/2006/05/026.

[20] B. G. Siddi and D. Müller. "Gaussino - a Gaudi-Based Core Simulation Framework" In: *2019 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)*. 2019, pp. 1-4.

[21] G. Barrand et al. "GAUDI — A software architecture and framework for building HEP data processing applications". In: *Journal of Computer Physics Communications* 140.1 (2001), pp. 45-55. ISSN: 0010-4655. DOI: https://doi.org/10.1016/S0010-4655(01)00254-5. URL:https://www.sciencedirect.com/science/article/pii/S0010465501002545

[22] G. James et al. "An Introduction to Statistical Learning: with Applications in R." In: Springer, 2017. Chap. 8, p. 336. ISBN: 978-1-4614-7137-0.

[23] L. Breiman et al. "Classification and Regression Trees". In: Wadsworth Publishing, 1983. ISBN: 9780412048418.

[24] R. Schapire. "The Boosting Approach to Machine Learning: An Overview". In: Springer, 2003. Chap. 8, pp. 149-171. ISBN: 978-0-387-21579-2.

[25] A. Natekin and A. Knoll. "Gradient Boosting Machines, A Tutorial". In: Frontiers in neurorobotics 7 (Dec. 2013), p. 21. doi: 10.3389/fnbot.2013.00021.

[26] M. Mazurek et al. *New simulation software technologies at the LHCb Experiment at CERN*. 2021. arXiv: 2112.04789[physics.ins-det]