laSalle ENG
Ramon Llull University

Escola Universitària d'Enginyeria Tècnica de Telecomunicació La Salle

Treball Final de Grau

Grau en Enginyeria Informàtica

Neutral particles identification in the LHCb calorimeter using Machine Learning techniques

Alumne
Xavier Roma Castells

Professor Ponent
Míriam Calvo Gómez

ACTA DE L'EXAMEN
DEL TREBALL FI DE CARRERA

Reunit el Tribunal qualificador en el dia de la data, l'alumne

D. Xavier Roma Castells

va exposar el seu Treball de Fi de Carrera, el qual va tractar sobre el tema següent:

Neutral particles identification in the LHCb calorimeter using Machine Learning techniques

Acabada l'exposició i contestades per part de l'alumne les objeccions formulades pels Srs. membres del tribunal, aquest valorà l'esmentat Treball amb la qualificació de

Barcelona,

VOCAL DEL TRIBUNAL VOCAL DEL TRIBUNAL

PRESIDENT DEL TRIBUNAL

Neutral particles identification in the LHCb calorimeter using Machine Learning techniques

by

Xavier Roma Castells

Computer Engineering

in

Ramon Llull University, La Salle BCN

May 2019

Abstract

Neutral particles identification in the LHCb calorimeter using Machine Learning techniques

by

Xavier Roma Castells

Computer Engineering

Ramon Llull University, La Salle BCN

Míriam Calvo Gómez, Doctor

Xavier Vilasís Cardona, Doctor

The LHCb restarts its activity at 2021 with an internal transformation allowing to run with a five times increase *pp* collision rate. Data used in this project comes from Monte Carlo simulation with the upgraded LHCb description.

Photons γ and neutral pions π^0 are identified by means of an electromagnetic calorimeter (ECAL). The classification performance γ/π^0 is studied during this work. Existing tools are replicated and compared with the ones developed using XGBoost and CNN techniques. Results from CNN perform better than the original tool and XGBoost method improves any previous implementations. Further increase in performance could be achieved by reducing cell size at the ECAL detector.

Key words
ECAL; XGBoost; CNN; LHCb; LHCb *upgrade*; CERN; LHC; Machine Learning; BDT.

Contents

List of Figures

List of Tables

Acknowledges

I would like to thank the collaborators of the research group for the support and help provided and that in a certain manner have contributed to this project. Specially my tutor, Dr. Míriam Calvo Gómez and my co-tutor Dr. Xavier Vilasís Cardona for the enormous help, their trust, dedication and great advises.

Finally to my friends and family specially to Gabri for the happiness and support behind those long hours of hard work.

Chapter 1

Introduction

This works studies new approaches to classify neutral particles in the ECAL detector at LHCb using data from energy deposits left by subatomic particles.

A brief introduction to LHC 1.1 and the Physics Standard Model 1.2 is done in this chapter.
Following the LHCb experiment is detailed all together with the different detectors 2.
In chapter 3, the ECAL detector is described with design features and challenges.

Chapter 4 explains how the data is obtained with Monte Carlo simulation and clarifies the distinct ways data is grouped all together with a definition of each individual variables found inside each aggregation.
Later in chapter 5, the dataset content is analysed as well as the hypothesis of studying energy deposits shape is exposed, also the normalisation and processing of the data is detailed.

In chapters 6 and 7 the theory behind the Machine Learning techniques employed as well as a look inside the implementation used in this project is illustrated.
Finally the results obtained are outlined and compared in chapter 8.
Last but not least, project conclusions are in chapter 9 alongside future lines.

1.1 The Large Hadron Collider at CERN

The Large Hadron Collider (LHC) at CERN is the world's most powerful tool for Particle Physics research[15]. LHC is 27km circumference ring tunnel located 100m below surface between France and Switzerland area, near Geneva.

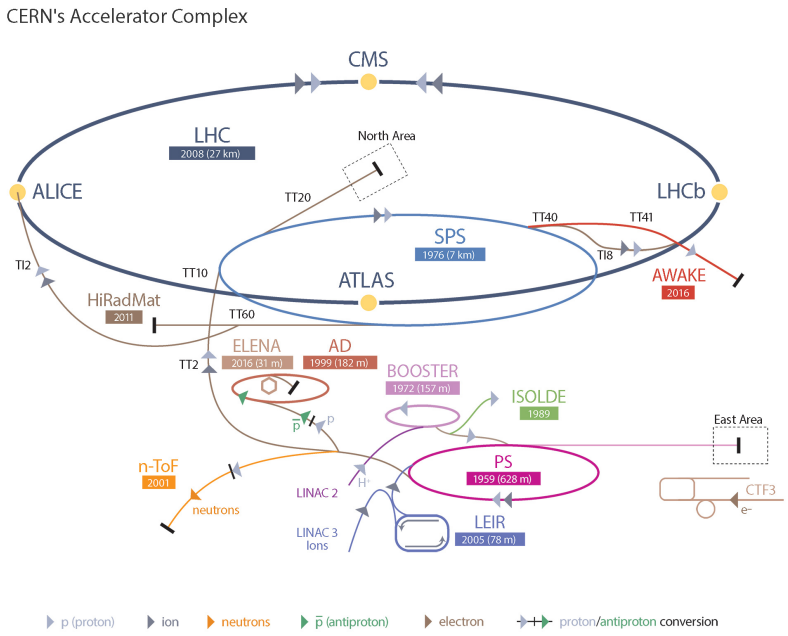


Figure 1.1: CERN Accelerator Complex. Protons are obtained by removing electrons from hydrogen atoms and they are first accelerated to 50 MeV by LINAC 2 (the LINear ACCelerator 2). The BOOSTER brings increases it's energy to 1.4 GeV and PS (Proton Synchrotron) brings them up to 26 GeV. SPS (Super Proton Synchrotron) accelerates them up to 450 GeV and finally the beams are divided and injected at LHC into two canals with opposite direction where they reach the 7 TeV energy.

LHC is designed to accelerate proton beams up to an energy of 7 TeV and collide them at 14 TeV. The beams are accelerated following certain steps, see figure 1.1 caption for step by step description.

Once the beams are injected at LHC ring, they are accelerated up to 0.999999991c velocity. To do so, very powerful and superconductors magnets are used to bend the protons and permit its correct circulation through the LHC ring. These magnets are kept at a temperature of -271 ° C, is below exterior space temperature. Once the beams are at maximum speed are allowed to collide at four Interaction Points (IPs) of the LHC ring. A collision between these protons might happen every 25ns, leading to a 40MHz collision rate.

LHC instantaneous luminosity corresponds to $10^{34} cm^{-2}/s$. These value is the key indicator of the accelerator performance, defined as the number of collisions per unit area over time.

Different experiments are placed at each LHC IP, which can be divided into two categories:

- General-Purpose Detectos (GPDs):

  - **ATLAS**, A Toroidal LHC ApparatuS.
  - **CMS**, Compact Muon Solenoid

  Both designed to study collisions producing high transverse momentum p_T particles. These GPDs confirmed the Higgs Boson. This discovery is highly sensitive to contribution from Beyond Standard Model (BSM) physics.

- Dedicated physics experiments, the main two are:

  - **ALICE**, A Large Ion Collider Experiment. Dedicated to the study of quark-gluon plasma (QGP) in heavy ion collisions.
  - **LHCb**, Large Hadron Collider beauty which studies c and b hadron decay. Further details are explained in 2.

1.2 Standard model

The Standard Model of particle physics (SM) and the general relativity (GR) are the two main theories used to describe the four known fundamental forces in the Universe as well as all known elementary particles. SM describes the electromagnetic, weak, and strong interactions, finally GR reports the gravitational force.

The Standard Model was introduced in the 70's, the experimental discoveries supporting the theory were observed decades later: the Higgs Boson elementary particle confirmed in 2012 by the ATLAS and CMS, for example. However, SM is not the ultimate theory of Nature. Several aspects of the basic structure of matter and cosmological observations can not be yet explained.

Following, the SM particle classification and the fundamental interactions will be briefly described.

1.2.1 Fundamental Particles Classification

The fundamental particles making up the SM have been shown experimentally to have no observed substructure down to the scale of $\approx 10^{-19}$m [25]. Each particle can be classified thanks to the "spin" i.e, a quantum property of particles, resulting in two sets: Fermions (half-integer spin particles) and Bosons (Full-integer spin particles).

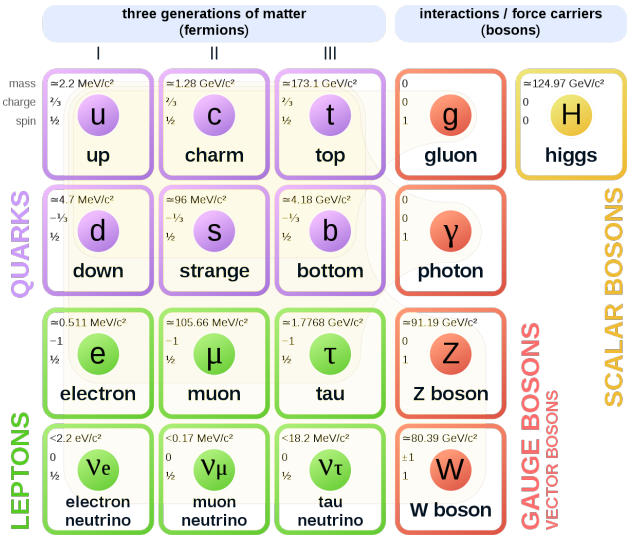


Figure 1.2: Standard Model of Elementary Particles.

1.2.1.1 Fermions

There are 12 elementary particles with 1/2 spin known as fermions, considered the basic building blocks of matter, each of it has an associated antiparticle. These particles can be divided into two groups according the way they interact with the strong interaction: quarks (up, down, charm, strange, top, bottom) and leptons (electron, electron neutrino, muon, muon neutrino, tau, tau neutrino).

1.2.1.2 Bosons

Bosons are defined as force carriers that mediate the strong, weak, and electromagnetic fundamental interactions. Like fermions, bosons can be divided into two groups: Vector Bosons with 1 spin and Scalar Bosons with 0 spin.

- The photon γ is the gauge boson of the electromagnetic force. Photons couple to all fermions except neutrinos, i.e. couple with electrically charged particles. The photon itself is mass-less,uncharged and does not decay into other particles or have any coupling to itself. This lack of self-coupling combined with the photon's zero mass results in the electromagnetic force having an infinite range for interactions.

- The weak force is mediated by the W and Z gauge bosons, which couple to all elementary fermions. The Weak interaction is responsible of natural radioactivity.

- The strong force is mediated by the g gluon. Gluons are mass-less and electrically neutral. As name hints, the strong force is by far the strongest of the fundamental

forces, approximately 102 times stronger than the electromagnetic force. As a result of its strength, gluons travel a short distance before interacting. The strong interaction is responsible of the binding and confinement of quarks inside hadrons.

- The last one, Higgs boson is in the scalar boson group. The Higgs boson was proposed to break its symmetry and give particles their masses.

1.2.1.3 Composite Particles

Composite particles are composed of two or more elementary particles.
Mesons are bosons composed of a quark and an antiquark. For example pions carry the nuclear force between nucleons: $\pi^0 = u\bar{u}/d\bar{d}$.

Baryons are fermions composed of three quarks. For example neutron: $n^0 = udd$.

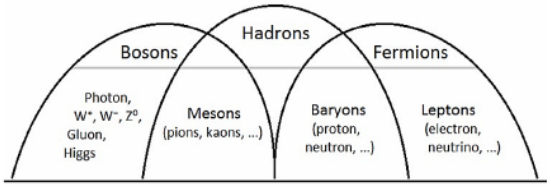


Figure 1.3: Combinations of Elementary Particles

Chapter 2

The LHCb experiment

LHCb stands for Large Hadron Collider beauty and is one of the main four. $b\bar{b}$ production in pp collisions are predominantly produced in the forward and backward direction, with respect to the proton beam direction, the LHCb detector is designed as a single arm forward spectrometer, see figure 2.1.

The LHCb detector has a polar angular coverage from 10 to 300 milliradians (mrad) in the horizontal and 10 to 250 *mrad* in the vertical plane, this allows to capture 27% of the b and $\bar{b}$ quarks produced.

Analyses of events containing useful information to explain matter and antimatter asymmetry rely on accurate measurements. Therefore the LHCb is composed of several sub-detectors, each of these are designed for specific purposes.

To push forward, LHC detectors are continuously improving with the latest technologies. Therefore complete stops are done approximately every 3 years, known as Long Shutdowns (LS). This shutdowns are aimed to improve and upgrade based on the experience obtained during the run and, of course, studies from the scientific community.

There has been two operational runs up to this date, between 2009 to 2013 and 2015 to 2018. At the time of writing this document, LHC is in LS period, which started at the end of 2018 and will retake at 2021.

This document does not describe the previous LHCb detectors, and detailed info can be found here [18]. Instead is focused on the upgrade design.

Figure 2.1 shows the different sub-detectors and how they are arranged. LHCb is LHCb sub-detectors groups and an insight of each sub-detector is described in the following sections.

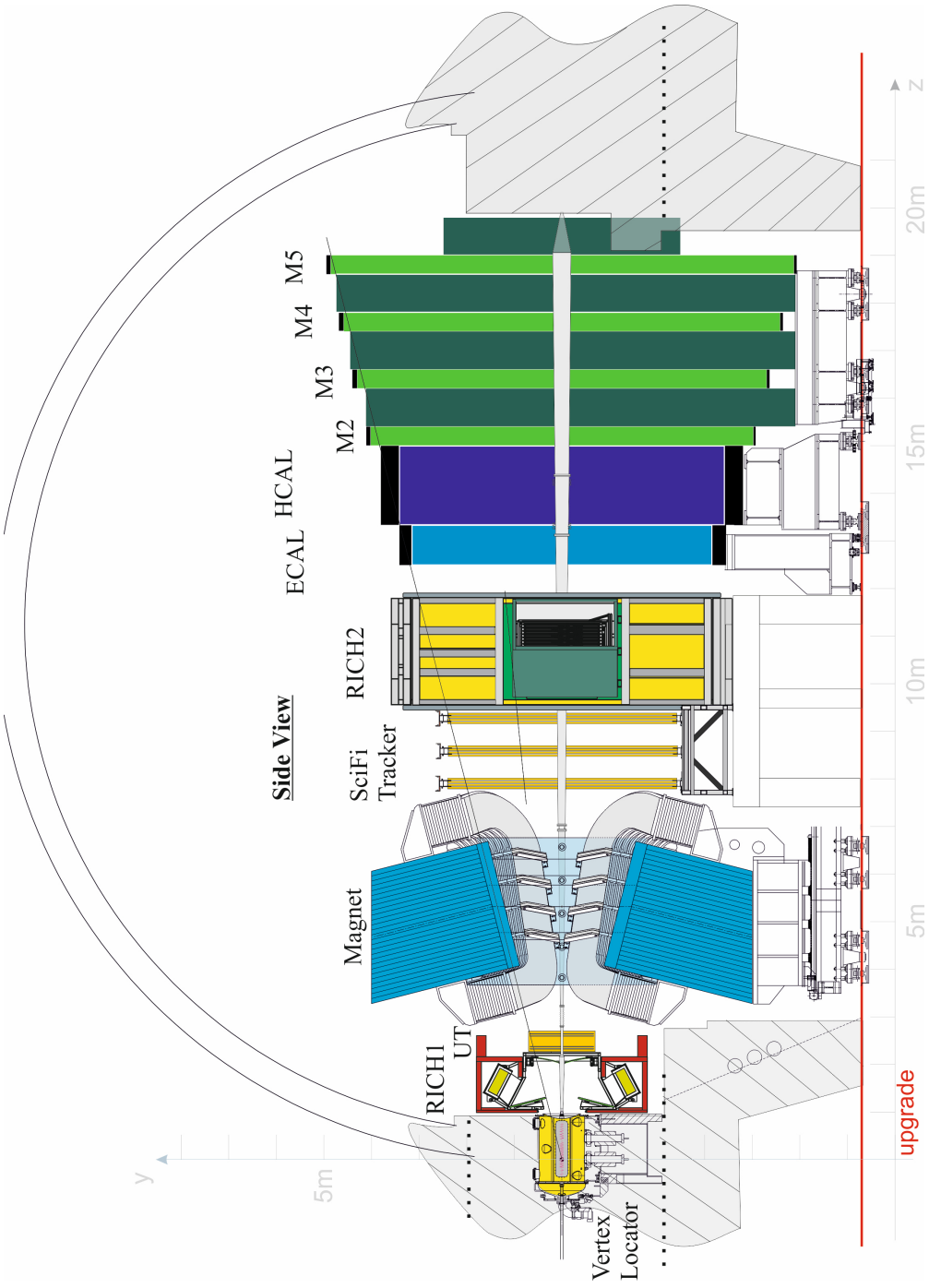


Figure 2.1: View of the upgraded LHCb detector.

2.1 The Tracking system

The Tracking system is composed of the VErtex LOcator (VELO), the Upstream Tracker (UP), the Dipole Magnet and the Scintillating Fibre Tracker (SciFi Tracker), the location of each individual detector is shown in figure 2.2.

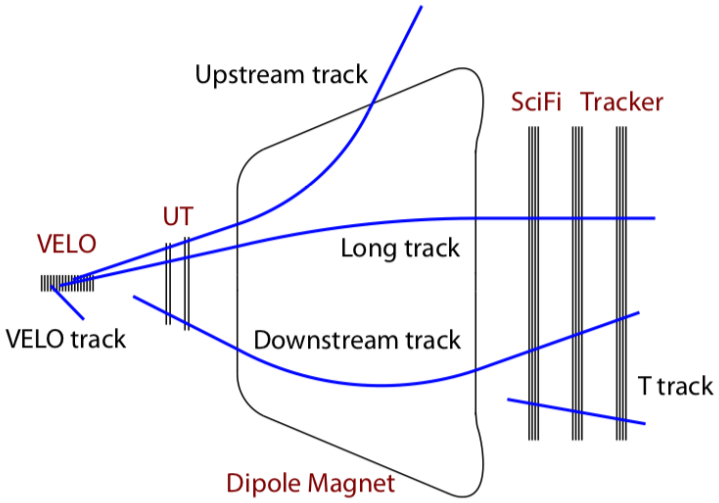


Figure 2.2: Locations of the detectors and reconstructed track types in the LHCb tracking system.

- The **Vertex Locator** job is to pick out B and D mesons, a tricky task since both decay close to beam. To find them, it must be positioned perilously close to the point where protons collide. To do so the silicon elements are moved mechanically towards or away from the beam. Moreover VELO measures the distance between the point where protons collide (and where B particles are created) and the point where the B particles decay, within 10 microns of error.

  The fact that VELO surrounds the area where the beams collide, permits identify the B and D mesons disintegration secondary vertex.

- The **Upstream Tracker** is just before dipole magnet and serves as a link between the VELO and T station trackers. It has four layers of silicon strips sensors. The middle layers are 5 and -5 degrees in the Y axis. UT is useful to extract information about the particles track directions.

- The **Dipole Magnet** is the following detector, it weights 16000 Tons and instead of being super-conducting it is a warm magnet design. This magnet extracts useful information about charged particles momenta, it does so measuring the amount bending

the particles suffer in its trajectory when these are under the magnetic field.

- Finally the **Scintillating Fibre Tracker** is composed by three stations: T1, T2 and T3. Each has four layers where the middle ones are slightly bent in the Y axis, like UT. The goal of this detector is to identify the entire track of each particle putting together the information of the stations.

2.2 The Particle identification system

The Particle identification system group the following systems.

2.2.1 Ring Imaging CHerenkov system

The Ring Imaging CHerenkov (**RICH**) system consists of an upstream detector (**RICH1**) positioned directly behind the VELO, and a downstream detector (**RICH2**) located behind the magnet and the SciFi Tracker. Both measure Cherenkov radiation which is an electromagnetic radiation emitted when a charged particle passes faster than light through that specific medium. Particles generate a light shock wave as it travels through the detector, RICH measures this wave to identify the speed that particle is travelling and afterwards this information helps identify the particle identity providing an hypothesis of its mass..

2.2.2 The Calorimeter System

The **Electromagnetic CALorimeter (ECAL)** system primary mission, as it names suggests, is to identify the energy and position of electromagnetic particles, such as electrons. The cell size varies from 4x4 cm^2 in the inner part of the detector, to 6x6 and 12x12 cm^2 in the middle and outer parts. The overall detector dimensions are 7.76 x 6.30 m^2. This system is further explained at chapter 3.

The **Hadronic CALorimeter (HCAL)** system is behind the ECAL and main goal is to measure the position and energy deposited by Hadrons, Hadrons leave a poor energy trace at ECAL. Similar to the ECAL, the 500 Tons detector inner and outer parts have different cell dimensions of 13x13 and 26x26 cm^2 , respectively. The overall dimensions are 8.40 x 6.80 m^2.

2.2.3 Muon Station System

The **Muon Station** System is composed of four stations (M2, M3, M4 and M5) of rectangular shape. Stations M2 to M5 are placed behind the HCAL. Station M1 is missing, it was used to improve the first level hardware trigger (Not present in the upgrade), and was placed in front of ECAL, see the gap present between RICH2 and ECAL at figure 2.1. This tracker is important since muons are present in the final state of many of the key decays.

Chapter 3

The Electromagnetic calorimeter

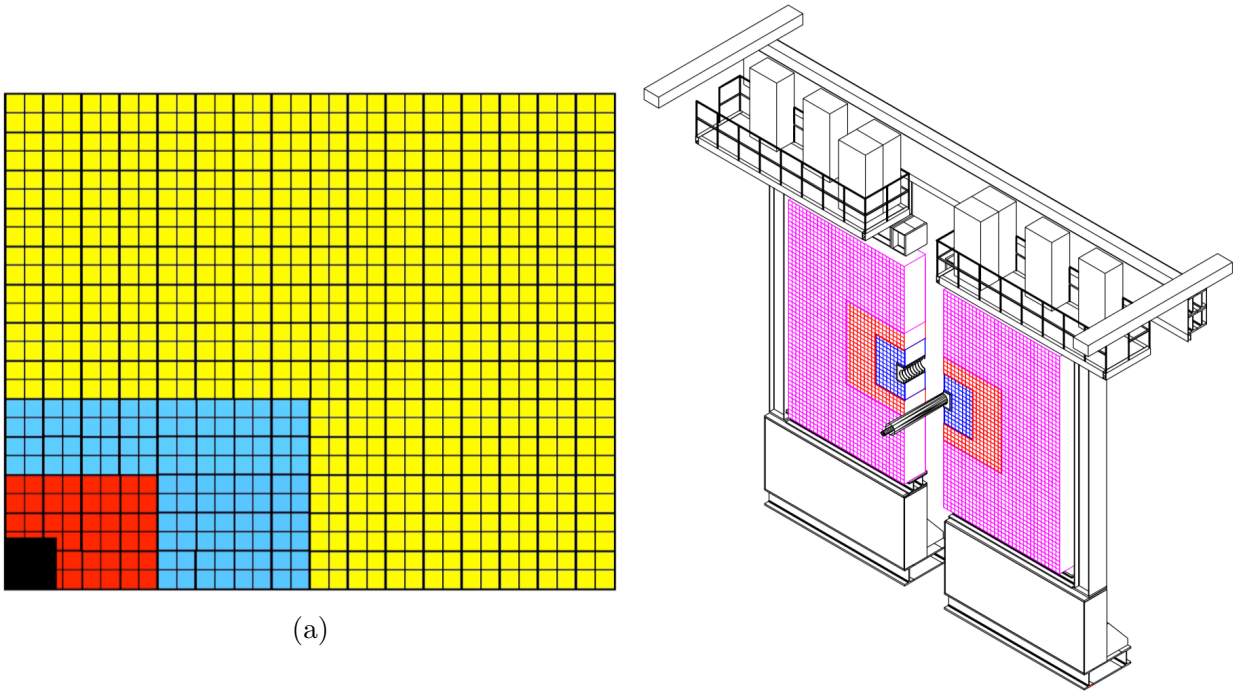
3.1 Overview

The Electromagnetic Calorimeter (ECAL) is a wall-like *shashlik* technology calorimeter. It is built from individual modules that consists of 66 layers of scintillators tiles (4mm) as an active material and lead absorber plates (2mm). Wavelength-shifting fibbers penetrate the layers through holes and are read by photo-multipliers at the back of the module.

The ECAL structure is divided into three sections, see figure 3.2a. All three sections have specific modules with identical square size of 121.2 mm but differ by the number of readout cells (figure 3.1). At section 3.2 further design details are detailed.

Figure 3.1: The figure shows the ECAL cell distribution per module: 1, 4, 9 readout cells per module corresponding outer, middle and inner sections respectively.

The ECAL starts at z=12.49 m from the interaction point and its 835 mm wide. The x



(a)

(b)

Figure 3.2: Separation between ECAL inner, middle and outer areas is shown in figure 3.2a. 3.2b Shows a 3d view from behind the detector. The three sections can be distinguished by its colour, the ECAL main platform, and the electronics platform with the racks on top of the ECAL wall. Around the beam pipe is drawn the inner supporting frame. One of two ECAL platforms is partially moved out. Figure taken from [22].

active area is 7.8 m and the y is 6.3 m, see figure 3.2b.

3.2 Design of modules

Table 3.1 describes the three types of module that build up the inner, middle and outer sections of the ECAL detector. All modules have lead absorber plates of identical size, but they differ by the number of cells and by the number of scintillating tiles per module, as well as by the fiber density, see figure 3.3.

The light from a scintillation tile is transported by 1.2 mm fibers to the back of the module where is read. The fibers that penetrate the lead scintillator stack are bent at the front of the stack, where loops are made, and at the rear of the stack, where fibers are formed to

	Inner Section	Middle Section	Outer Section
Inner size, $x \times y \times cm^2$	65 × 65	194 × 145	388 × 242
Outer size, $x \times y \times cm^2$	194 × 145	388 × 242	776 × 630
Cell size, cm^2	4.04 × 4.04	6.06 × 6.06	12.12 × 12.12
# of modules	176	448	2688
# of channels	1472	1792	2688
# of cells per module	9	4	1
# of fibers per module	144	144	64
Fiber density, cm^{-2}	0.98	0.98	0.44

Table 3.1: Main parameters of the LHCb electro-magnetic calorimeter

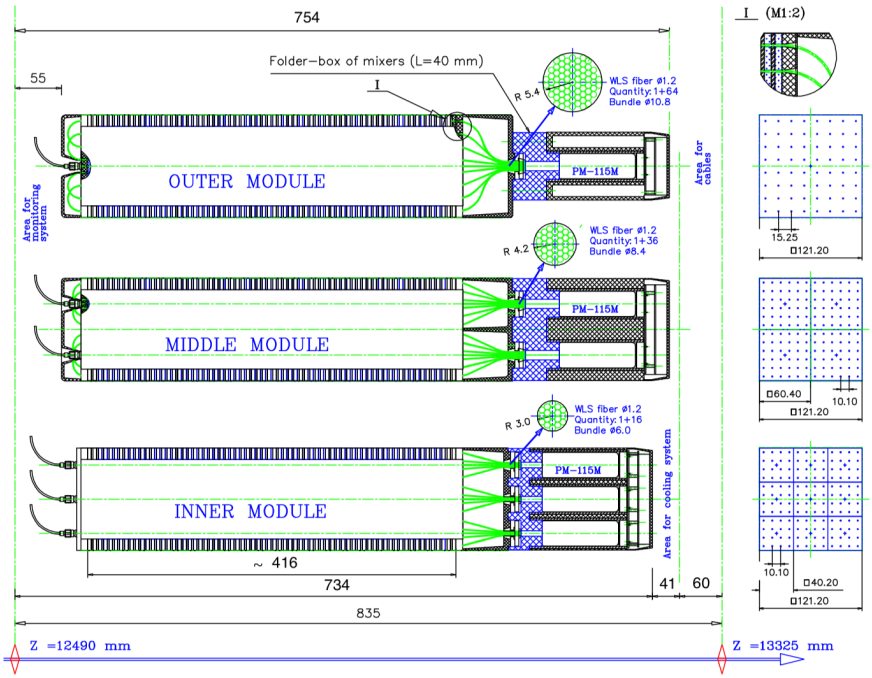


Figure 3.3: The ECAL modules for the different sections. At the monitoring side (left side) the fibers, connectors, fiber loops and plastic covers are shown. At the readout side, the fiber bundles, the light mixers, the photo-multiplier tube (for reading purpose), and their bases are shown.

a bundle. To avoid light loss caused by the bent, a special technique of fiber bending under extreme temperature air has been elaborated and successfully applied.

For the inner section modules, the bending radius would be very small, therefore and to avoid mechanical damage the fibers are cut and coated with an aluminium mirror.

3.3 Electromagnetic calorimeter challenges

The main purposes of the ECAL in the LHCb experiment are [4]:

- To reconstruct photons with a precision sufficient for the reconstruction with good mass resolution of B-meson decays including γs or π^0s.
- To participate in particle identification, particularly of electrons.
- Previously, before the upgrade during the LS2. It was used to provide energetic cluster candidates for the zero level L0 trigger that selected b-containing events.

ECAL does also provide a fast response within 25 ns, and a good reliability to keep operating after decades in a radiation environment.

ECAL total cost, taking into account realistic labor prices in different countries was 7M Eur

Chapter 4

Dataset description

In this chapter, the data set used to study the performance of different Machine Learning Classification algorithms will be discussed. First Monte Carlo simulation will be briefly explained. Following the used data with different attributes will be reviewed, as well as a close look up into the different classes of it.

4.1 The Monte Carlo Simulation

Monte Carlo simulation (MC) is a program that provides predictions of the behaviour of collisions between particles within the LHC detector. Its purpose is to generate, as detailed as possible, the detector response given simulated collisions.

The data generated in these simulations is used to test and verify the detector's behaviour against real collisions and to help physicists understand the behaviour of the particles observed in the results. It is also used as an input to the algorithms, which have the advantage that can be trained with so-called "MC truth". Therefore, the MC simulation plays an

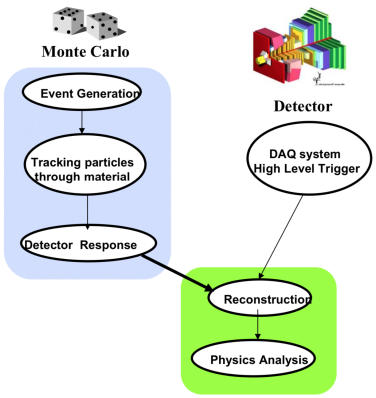


Figure 4.1: Usage of MC and LHCb Detector information

important role in the experiment. Comparing the simulated data with what is measured in the reality allows a better understanding of the results, see figure 4.1.

The process of generating an MC event is divided into two different simulation programs. The first is called *Gauss Simulation*. Gauss goal is to generate primary collisions and simulate the interaction of them on the detector. It is also responsible for determining which particles are created in a given particle collision based on the probability determined by the SM, while maintaining some randomness like present in Nature.

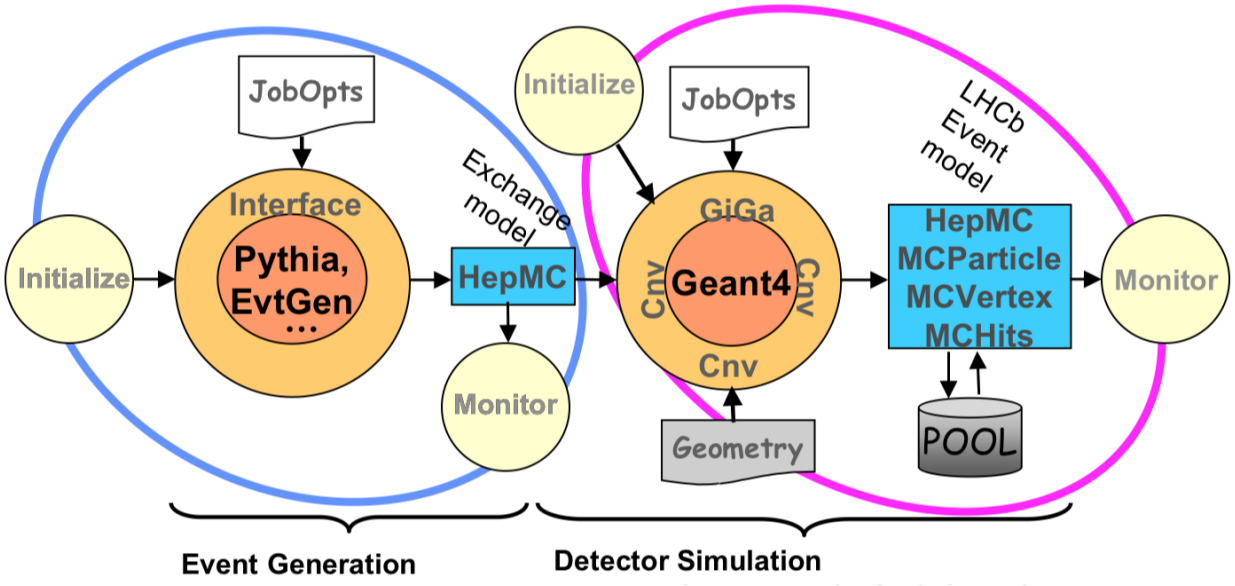


Figure 4.2: Structure of a Gauss application

Due to the complexity of the different LHC detectors, many MC simulations are dedicated to specific areas that together provide a global view of the event. Therefore, there are different programs dedicated to specialised simulations such as decays or hadronization, as well as general purpose programs: available in the physics community some examples are ISAJET or SHERPA. Experiments usually use one generator for massive production, and then make smaller data sets with the desired collision particularities needed.

The program *Geant4*, needs to describe the geometry of the detector in great detail in order to retrieve results as close as possible.

The second part of the simulation, called *Boole Digitalization*, is responsible for converting the information generated by the Gaussian simulation into the output data of the detector sensor. Thus, the digitised analogue information has the same output format as the same simulated event would have in the real detector, this allows direct comparison of the results.

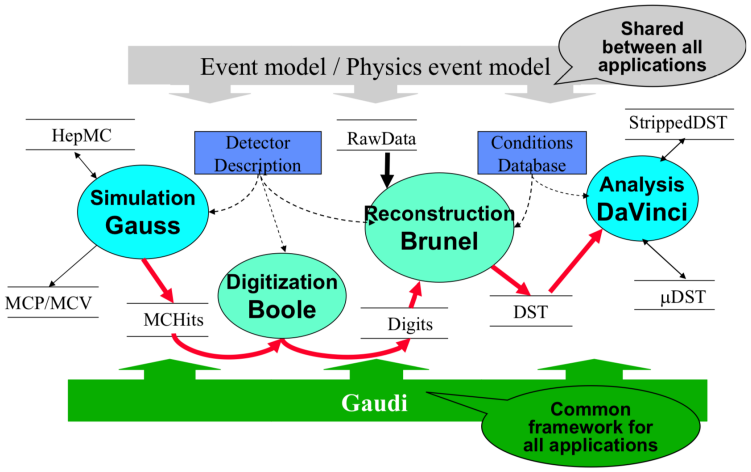


Figure 4.3: Flow of simulated data and applications

Once MC has generated the particles tracks, as well as all the information associated to each particle, and the detector response with the same output as the it would produce. The data is ready to be used for training algorithms, to evaluate the detector performance in order to improve it, etc.

Further processes are applied to the generated data in order to evaluate physics reach, those details are not discussed as are not in the scope of this project. See figure 4.3 or [13] for further details.

4.2 Dataset description

The data used contains both single photons and pairs of photons. To obtain particles, the following process is followed:

- For photons, the events that come from the simulation of $B^0 \to K^{*0}\gamma$ decay are used. Then check if the γ that comes from the B^0 decay has been reconstructed as a neutral cluster in the ECAL. A neutral cluster means that the tracking system does not have any charged particle pointing the cluster, see figure 4.4. The clusters are reconstructed using a cellular automaton.

- For pair of photons, the events used are those that come from the simulation of $B^0 \to \pi\pi\pi^0$ decay. The pair of photons comes from the π^0 decay, each photon goes to the same reconstructed cluster and both result in the same or an adjacent cell.

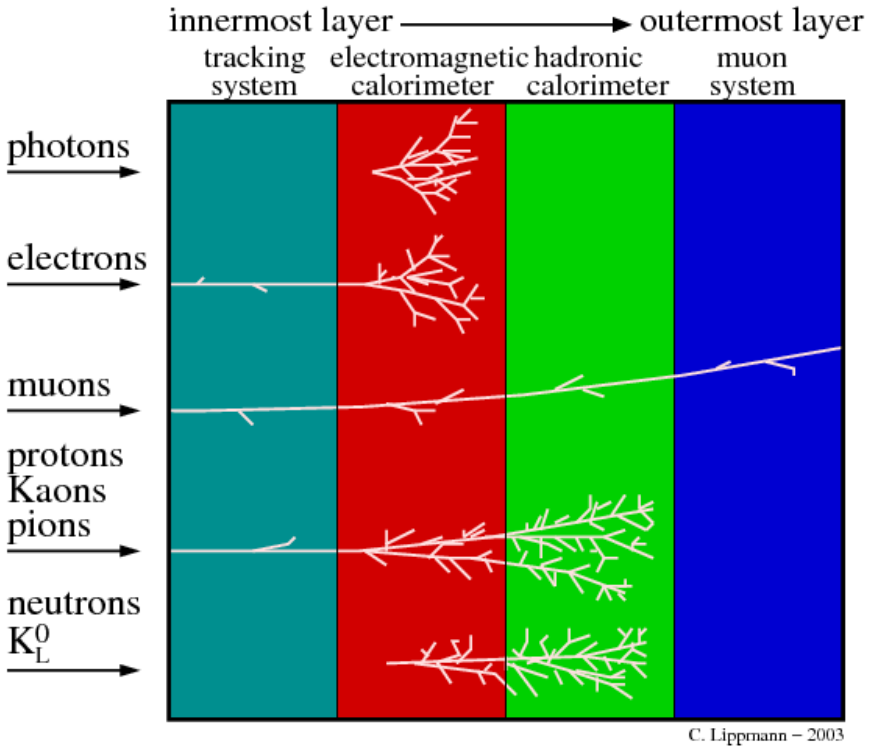


Figure 4.4: Trace left in the different detectors per particles.

4.2.1 Rows

A row of the dataset used contains the following fields:

- **p**: Particle momentum, the product of the mass and velocity of a particle.
- **pt**: The component of momentum transverse (i.e. perpendicular) to the beam line.
- **eta**: pseudo-rapidity, describes the angle of a particle relative to the beam axis.
- **area**: ECAL area identifier, 0 for the outer, 1 for the middle and 2 for the inner.
- **nPVs**: Number of reconstructed collisions in PV, the greater the value the more cluster overlapping in the ECAL.
- **ClusterE**: Energy of the reconstructed cluster, as clusters from this dataset are 3x3 cells, since the original cluster from the particle could be bigger than 3x3.
- **ClusterE9**: Energy of the 3x3 cluster. The energy outside (EnergyO) the cluster is

$$EnergyO = ClusterE - ClusterE9$$

. This value should validate the following sum

$$ClusterE9 = \sum_{i=1}^{9} ClE_i$$

.

- **DigEi**: From now on, referred as **Raw Digits Data**. It is the energy per cell like the ECAL output is, where i is the cell and ranges from 1 to 25.

DigE21	DigE22	DigE23	DigE24	DigE25
DigE16	DigE17	DigE18	DigE19	DigE20
DigE11	DigE12	DigE13	DigE14	DigE15
DigE6	DigE7	DigE8	DigE9	DigE10
DigE1	DigE2	DigE3	DigE4	DigE5

Figure 4.5: The cell id from the ECAL cluster data: $DigE_i$

- **ClEi**: From now on, referred as **Raw Cluster Data**. It is the energy per cell from the reconstructed cluster using a cellular automaton, where i is the cell and ranges from 1 to 25.

ClE21	ClE22	ClE23	ClE24	ClE25
ClE16	ClE17	ClE18	ClE19	ClE20
ClE11	ClE12	ClE13	ClE14	ClE15
ClE6	ClE7	ClE8	ClE9	ClE10
ClE1	ClE2	ClE3	ClE4	ClE5

Figure 4.6: The cell id from the reconstructed cluster data: ClE_i

- From now on referred as **Pre-Original Data**:
  - **Sxx**: Tell the spread of the energy through the X axis.
  - **Sxy**: Gives the spread of the energy through the diagonal inferred from the Y and X axis.
  - **Syy**: Provides the spread of the energy through the Y axis.

- From now on referred as **Original Data**:
  - **isPhr2**: Related to the size of the cluster.
  - **isPhr2r4**: Informs about the importance of the tails.
  - **isPhasym**: Provides information about the orientation of the ellipse or correlation between X and Y coordinates.
  - **isPhkappa**: It relates the major and minor semiaxes of an ellipse, it is a ratio between the eigenvalues (characteristic values) of the matrix S.
  - **isPhEseed**: Is the ratio between the seed cell and the clusters energy.
  - **isPhE2**: The importance of the aggregated seed cell and the second most energetic cell energy related the cluster energy.

  Further information about Original and Pre-original Data variables as well as detailed process to obtain them can be found here [6].

Chapter 5

Data Analysis

Separation between π^0/γ is an important prerequisite for the study of radioactive decays or B decay models with a π^0 in the final state. In order to reduce physics background containing π^0 or instead rejecting photons to study both cases. Therefore the study and development of powerful tools with good π^0/γ separation performance is an important task.

The method to obtain this dataset and the different attributes has been explained in section 4.2. In this chapter, an exhaustive look up to the dataset is done. The hypothesis to evaluate raw energy deposits is explained. Moreover the processing methods applied and the reason of them is discussed.

5.1 Raw Data

The hypothesis to evaluate energy deposits comes from the appreciable difference left by π^0 and γ at the ECAL. In the following subsection the noticeble variation is shown. First *raw digits* data is inspected, following raw cluster data is analysed.

5.1.1 Raw Digits Data

As seen in figure 5.1, single hits of π^0 differ in their shape compared to γ in the same ECAL area. π^0 hits tend to leave more energy in different cells as the energy deposited comes from a pair of γ. Instead γ tend to leave their energy in a single cell.

Taking a closer look at figure 5.1, γ hits (right row of the graphs in the figure) affect smaller areas and leave less energy than $\gamma\gamma$ hits. These (left row of the graphs in the mentioned figure) tend to spread their energy in broader way and at the same time the amount of energy left is significantly greater, see the plot warmer colour in π^0 graphs compared to γ ones.

Figure 5.2 compares all the accumulated hits per ECAL area and particle. Although the plot colour is warmer in π^0 graphs, is hardly appreciable and both seam to be the have similar values. The need to pre-process data takes importance.

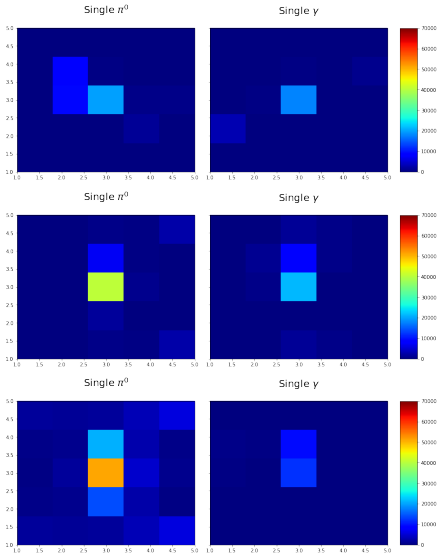


Figure 5.1: Unprocessed 5x5 Clusters from a single π^0 (at the left) and γ (at the Right) hit, at the ECAL outer, middle and inner area, top to down respectively.

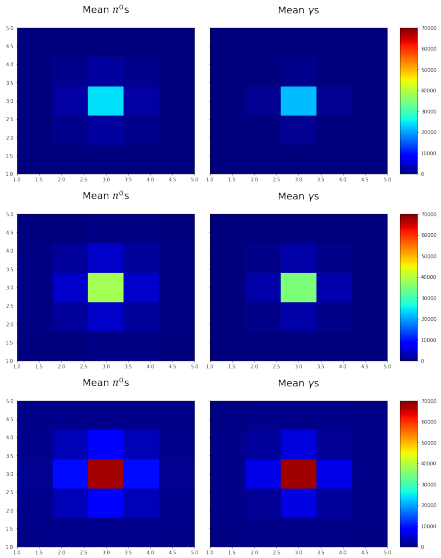


Figure 5.2: Unprocessed 5x5 Clusters mean of all π^0 (at the left) and γ (at the Right) hits, at the ECAL outer, middle and inner area, top to down respectively.

5.1.2 Raw Clusters Data

Figure 5.10 evidences even further the differences seen in the previous section, here data is obtained using a cellular automaton over the *raw digits* data, hence "only" the specific particle energy is kept.

Clear differences between 5.1 and 5.3 figures can be quickly identified, 5.1 have more energy in the outermost cells, this is due the overlapping between particles in the ECAL. Instead 5.3 do no longer have it. This difference is also appreciable in the averages of all hits in charts 5.2 and 5.4 where in the last, the outermost cells do not have nearly anything of energy in comparison to the innermost ones.

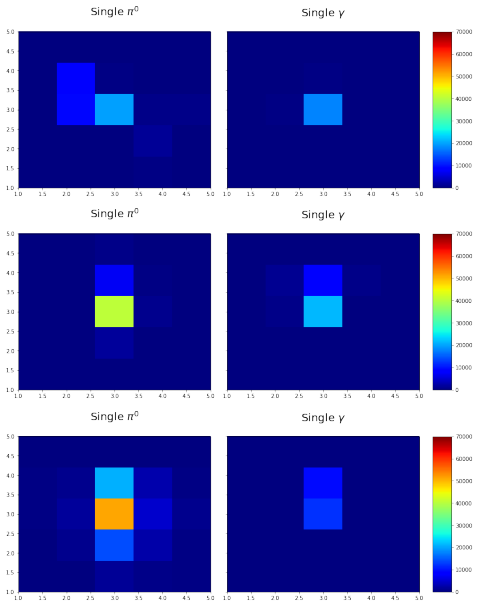


Figure 5.3: Unprocessed 5x5 Clusters from a single π^0 (at the left) and γ (at the Right) hit, at the ECAL outer, middle and inner area, top to down respectively.

5.1.3 Imbalance between classes

The number of samples per classes is compared in figure 5.5. Training ML algorithms with this unbalanced class samples could infer into always predicting γ, there is near 5 times more specimens. Therefore the need to balance them is important.

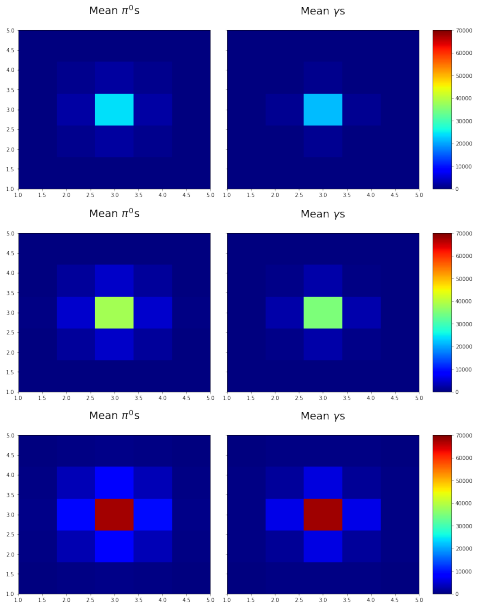


Figure 5.4: Unprocessed 5x5 Clusters mean of all π^0 (at the left) and γ (at the Right) hits, at the ECAL outer, middle and inner area, top to down respectively.

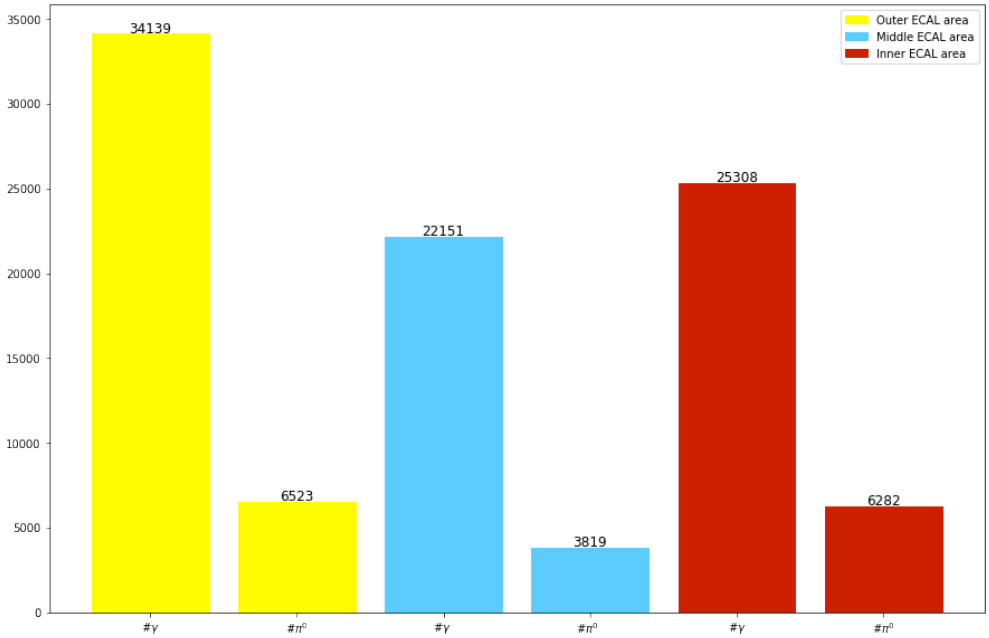


Figure 5.5: Number of samples per particle π^0/γ and classified per area. Overall γ samples is almost 4.5 times greater than π^0.

5.2 Processing data

The clusters of our classes have to be analysed by the shape left by the energy, not the amount of energy. Therefore data is normalised, the ratio between the entire energy of the 5x5 cluster and each cell:

$$Norm\_ClE_i = \frac{ClE_i}{\sum_{j=0}^{n} ClE_j} \mid Norm\_DigE_i = \frac{DigE_i}{\sum_{j=0}^{n} DigE_j}$$

Where N is the number of cells, 25 for 5x5 clusters and 9 for 3x3 ones.

Moreover, in order to avoid learning from undesired features, clusters are rotated. The rotation meets the following statement: the adjacent cell to the seed cell with greater energy will always be in the north or in the north-west, this means data, represented as a matrice, will be rotated. Beaware that the cell in the north and in the north-west of the seed cell are *Cl/DigE18* and *Cl/DigE17* respectively 5.6.

ClE21	ClE22	ClE23	ClE24	ClE25
ClE16	ClE17	ClE18	ClE19	ClE20
ClE11	ClE12	ClE13	ClE14	ClE15
ClE6	ClE7	ClE8	ClE9	ClE10
ClE1	ClE2	ClE3	ClE4	ClE5

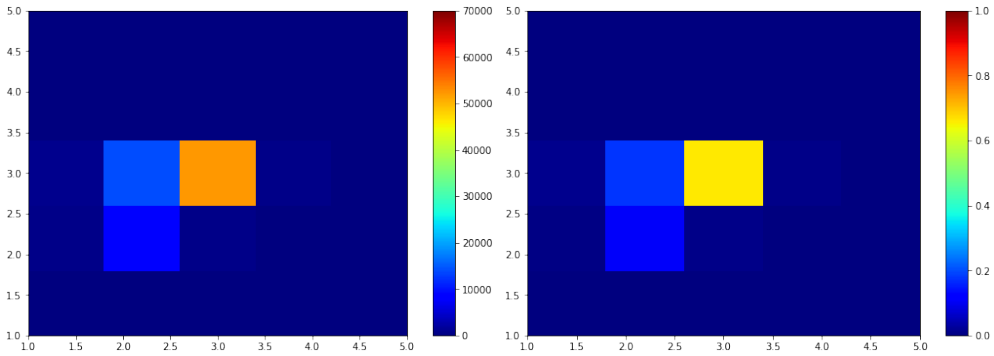
DigE21	DigE22	DigE23	DigE24	DigE25
DigE16	DigE17	DigE18	DigE19	DigE20
DigE11	DigE12	DigE13	DigE14	DigE15
DigE6	DigE7	DigE8	DigE9	DigE10
DigE1	DigE2	DigE3	DigE4	DigE5

Figure 5.6: Organisation of the cells ids in a cluster.

This process can be considered as another step of normalisation, since this method tries to avoid learning from such features, where in the dataset π^0 could locate the greater adjacent cell on the bottom with respect the seed cell. Therefore the training could learn about this undesired feature.

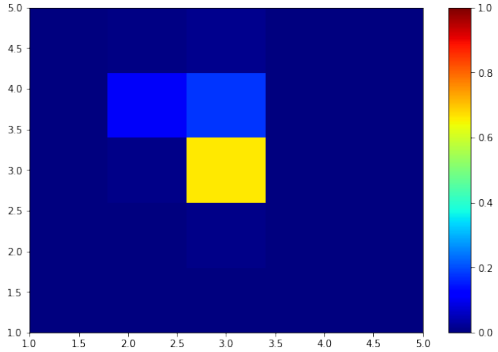
The following figure 5.7 enhance the steps followed.

Figures 5.8 and 5.9 show the result after the process applied at 5.7 and explained in this section for *DigEi clusters*. In an analogous way, figures 5.10 and 5.11 shows the same for *ClEi clusters*. The same samples exhbited in this figures can be compared to their initial states in 5.1 for 5.8, 5.2 for 5.9, 5.3 for 5.10 and 5.4 for 5.11. Note the difference before and after the operation and that the hypothesis above mentioned takes now greater importance. Figure 5.11 evidences that π^0 hits deposit energy in a wider area compared to γ ones.



(a) Raw π^0 hit cluster.

(b) Raw π^0 hit cluster normalised.



(c) Raw π^0 hit cluster normalised and rotated.

Figure 5.7: A single random merged π^0 present in the dataset looks like 5.7a, where each cell is the energy left by a $\gamma\gamma$ hit at the ECAL wall, the cluster is centred so the cell in the middle is where the major energy is left. The first step it is to normalise the cluster, the ratio between the entire energy of the cluster and each individual cell transforms it to 5.7b. Finally is rotated to avoid learning from bad features, resulting in 5.7c.

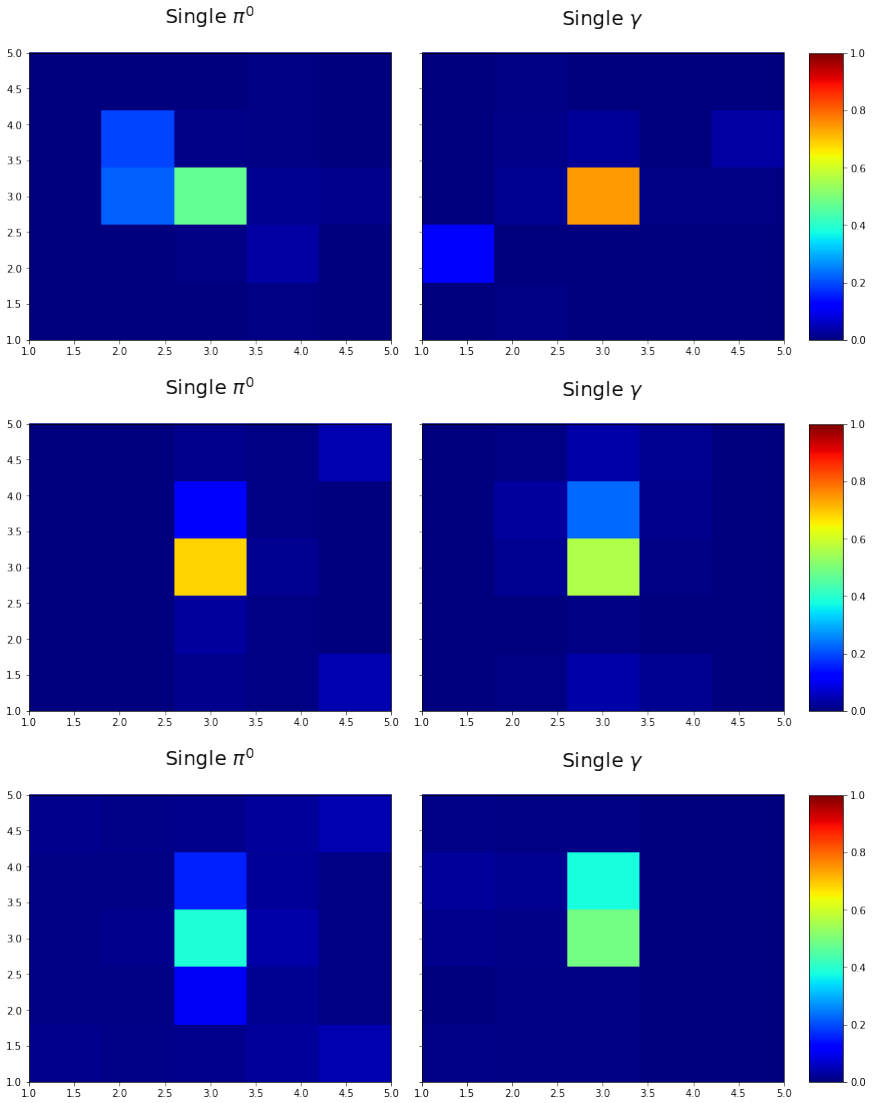


Figure 5.8: *5x5 DigEi Clusters* from a single π^0 (at the left) and γ (at the Right) hit, at the ECAL outer, middle and inner area, top to down respectively after being processed. The same clusters previous to refinement can be seen in figure 5.1.

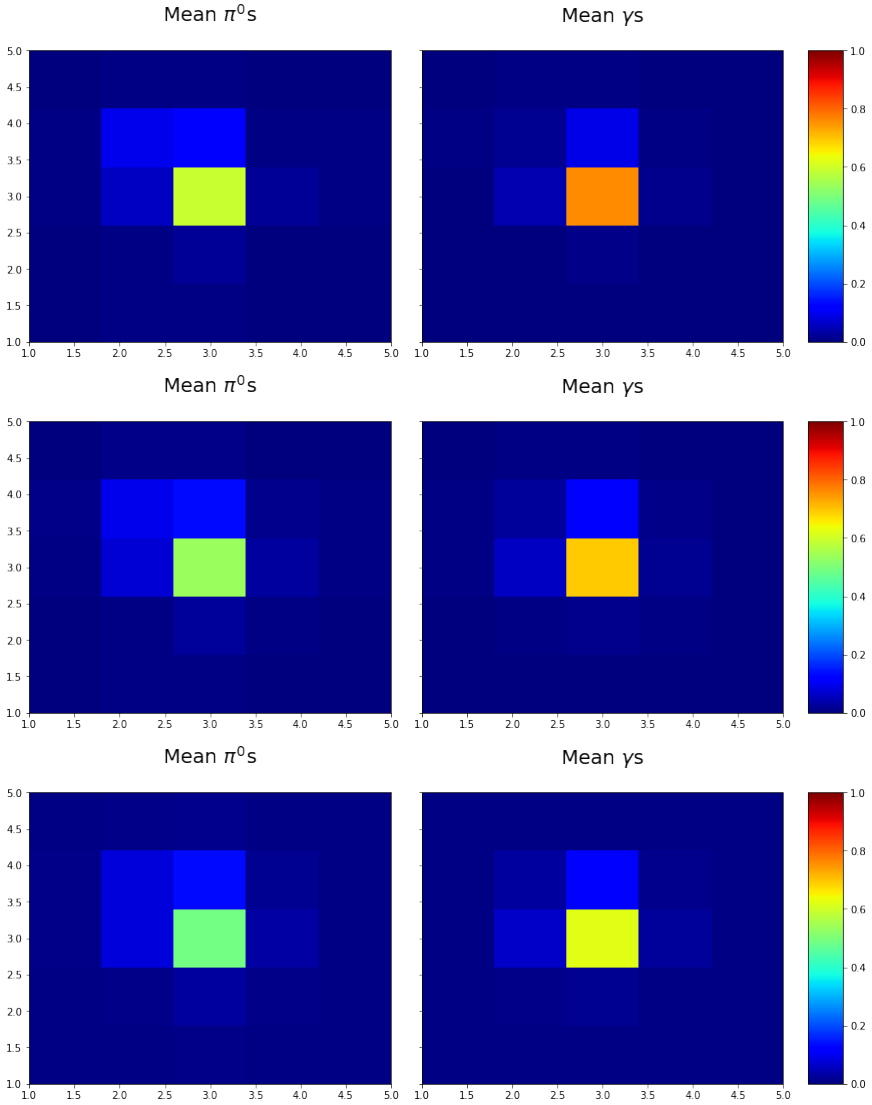


Figure 5.9: *5x5 DigEi Clusters* resulting from processing them. Average of all π^0 (at the left) and γ (at the Right) clusters present in the dataset, at the ECAL outer, middle and inner area, top to down respectively. See figure 5.2 for same figures without being refined.

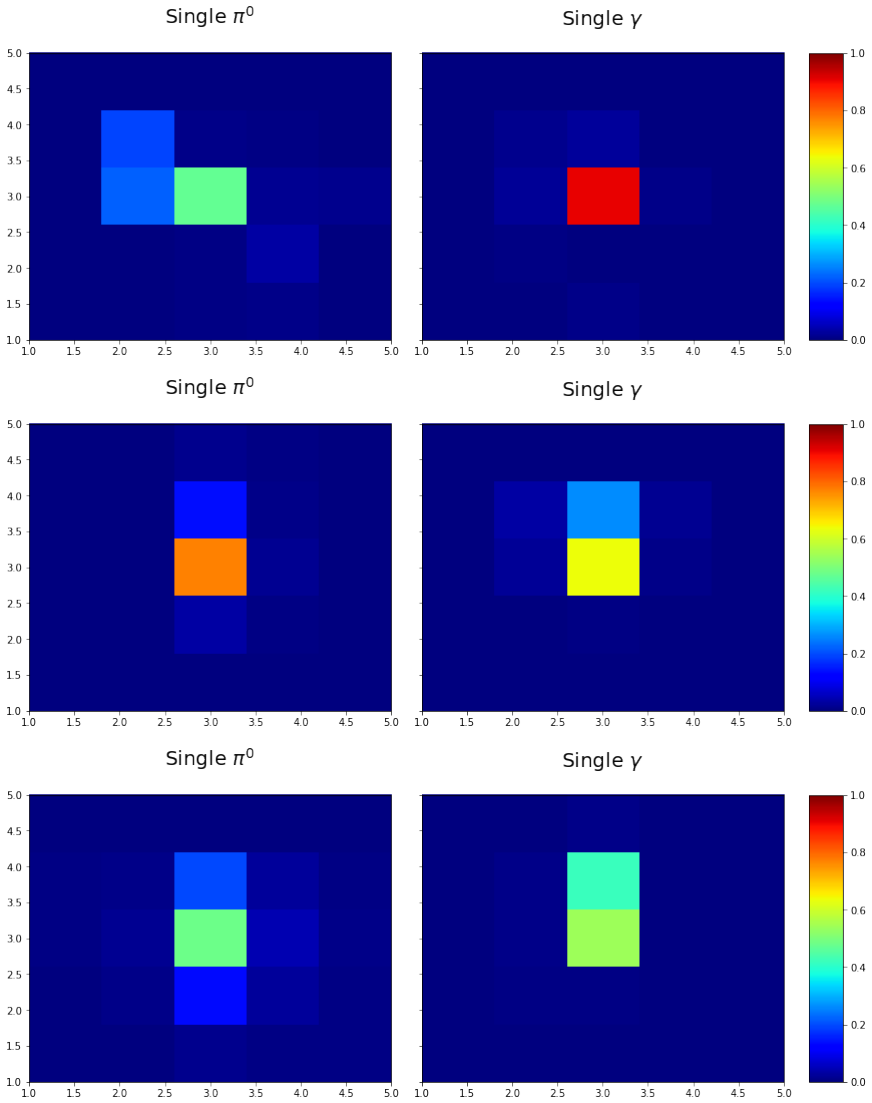


Figure 5.10: *5x5 ClEi Clusters* from a single π^0 (at the left) and γ (at the Right) hit, at the ECAL outer, middle and inner area, top to down respectively after being processed. The same clusters previous to refinement can be seen in figure 5.3.

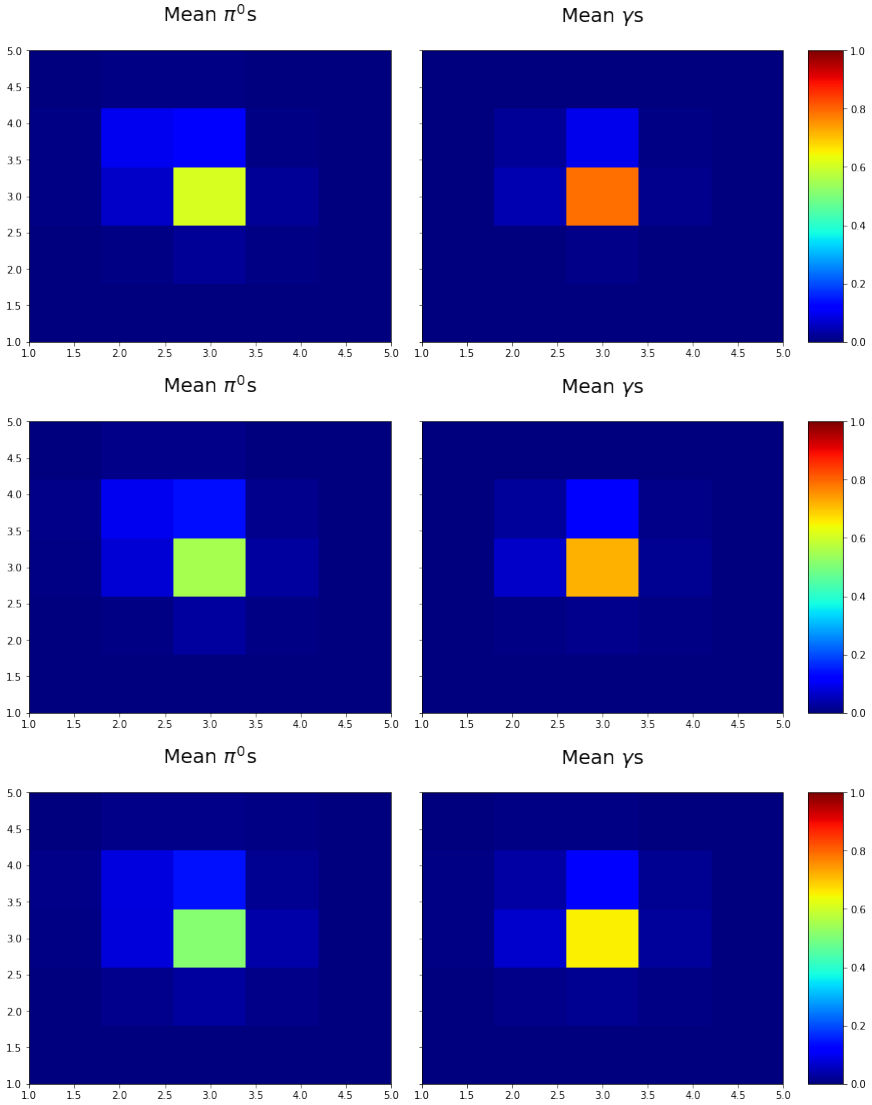


Figure 5.11: *5x5 ClEi Clusters* resulting from processing them. Average of all π^0 (at the left) and γ (at the Right) clusters present in the dataset, at the ECAL outer, middle and inner area, top to down respectively. See figure 5.4 for same figures without being refined.

Chapter 6

XGBoost

6.1 Classification Trees

A decision tree is a predictive model which can be used for both classification and regression problems. When a decision tree is used for classification tasks, it is referred as a classification tree, in the other hand, when is used for regression tasks, it is called regression tree.

Classification Trees are used to classify (in this work a cluster) to a predefined set of classes (π^0/γ) based on their attributes values (such as shape of the cluster or energy distribution).

Moreover simple classification trees are usually represented graphically, making them easier to interpret than other techniques, see figure 6.1.

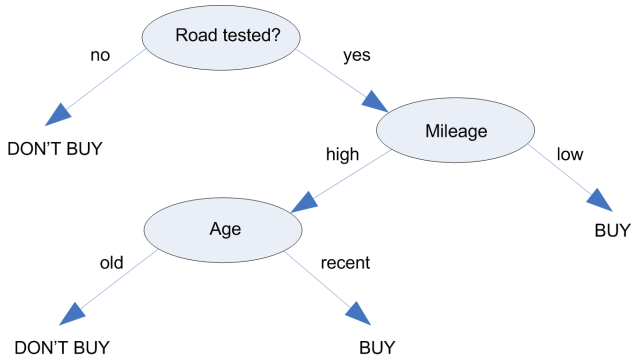


Figure 6.1: Simple decision tree for buying a car.

6.1.1 Characteristics of classification trees

The tree structure consists of nodes, root node and internal nodes, and leafs also known as terminal or decision nodes.

Each internal node splits data into two or more sub-datasets according to a certain discrete function of the input attributes value.
Each leaf contains a class or a probability vector, indicating the probability of the target class represented as a number.

6.2 Supervised learning

Supervised learning problems are those where training data (with multiple features) is used to predict a target variable. The final goal is to approximate the mapping function, so when new input data is introduced it can be predicted correctly.

The name supervised learning comes from the learning process of the algorithm, where there exists a training dataset that supervises the learning, while the algorithm in an iterative way, makes predictions and those are corrected with the training dataset, which contains the "truth".

The elements introduced below form the basic elements of supervised learning, and they are natural building blocks of machine learning.

- The **model** refers to the mathematical formula of which a prediction y_i is made from the input x_i. For instance, a *linear model*, the prediction is given as a linear combination of weighted features:
$$y_i = \sum_j \theta_j x_j^i$$
- The **parameters** (θ in the previous linear model example) are the part that the algorithm needs to learn from the data. Parameters are usually represented with θ.
- The **objective function** measures how well the model fit the training data.
$$Obj(\theta) = L(\theta) + \Omega(\theta)$$
It consists of two parts:
  - **Training loss** (L) measures how well does the model predict with respect to the training dataset. Optimising training lost encourages predictive models. For example a widely used L is the *mean squared error*:
$$L(\theta) = \sum_i (y_i - y_i')^2$$
  - **Regularisation term** (Ω) controls the complexity of the model It helps avoid model overfitting. Optimising regularisation encourages simple models. Simpler models tend to have smaller variance in future predictions, making prediction stable.

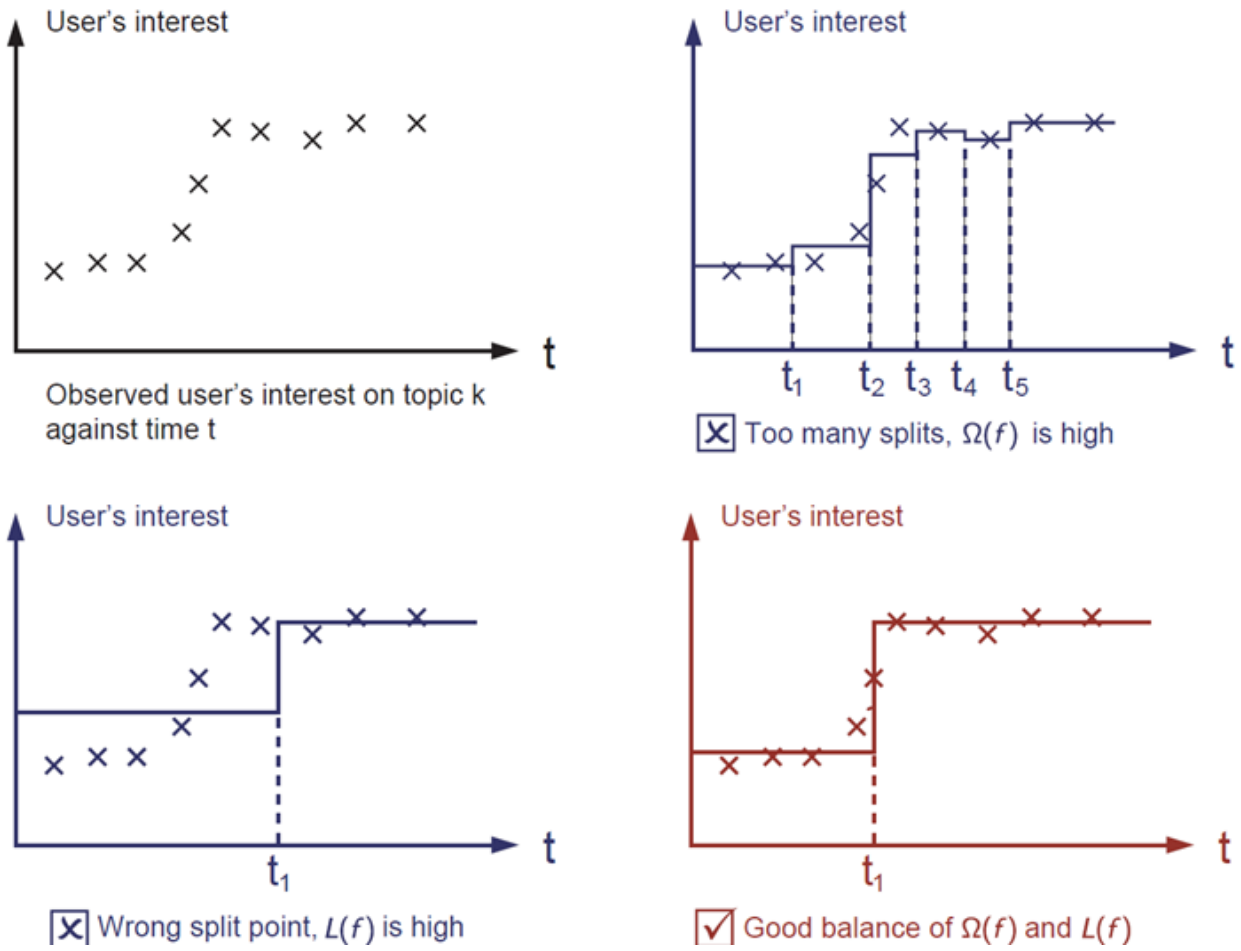


Figure 6.2: See how complex models where (Ω) is high induce to overfitting. Also note that models with big (L) will make predictions be poor even with the training dataset. Therefore, in red the a good tradeoff between training loss (L) and regularisation (Ω) is the key. Figure from [10]

The tradeoff between Training loss and Regularisation is known as *bias-variance trade-off*, see figure 6.2.

6.3 Decision Tree Ensembles

XGBoost is a model that consists of a set of Classification And Regression Trees (CART). CART are characterised by the fact that they construct binary trees, i.e, each node has only two outgoing paths. The splits are selected using the Twoing Criteria and the obtained tree is pruned by Cost-Complexity Pruning.

In figure 6.3, a score is assigned on the corresponding tree leaf. A CART makes decision rules like a decision tree, but has a main difference: leafs only contains decision values. In CART, a real score is associated with each of the leafs, which gives richer interpretations that go beyond classification.

Tree Ensembles is a technique that uses multiple trees (a set of CARTs), and sums the prediction of all the trees together making the predictions more robust. Figure 6.4 is an example of a two tree ensemble. Where the prediction score of each tree is added to output the final prediction score.

Mathematically the prediction of a tree ensemble model with K trees and $\mathcal{F}$ is the space of functions containing all CART trees:

$$y_i' = \sum_{k=1}^{K} f_k(x_i), f_k \in \mathcal{F}$$

And the objective function to be optimised is given by:

$$Obj(\theta) = \sum_{i}^{n} L(y_i, y_i') + \sum_{k=1}^{K} \Omega(f_k)$$

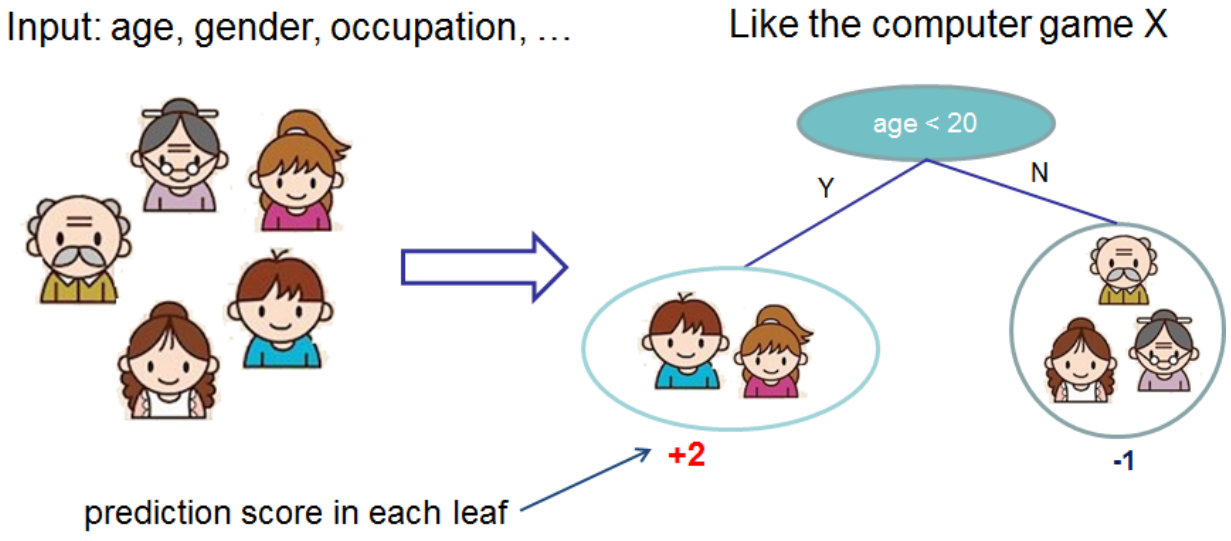


Figure 6.3: Simple example of a CART that classifies whether someone will like a hypothetical computer game X. Figure from [10]

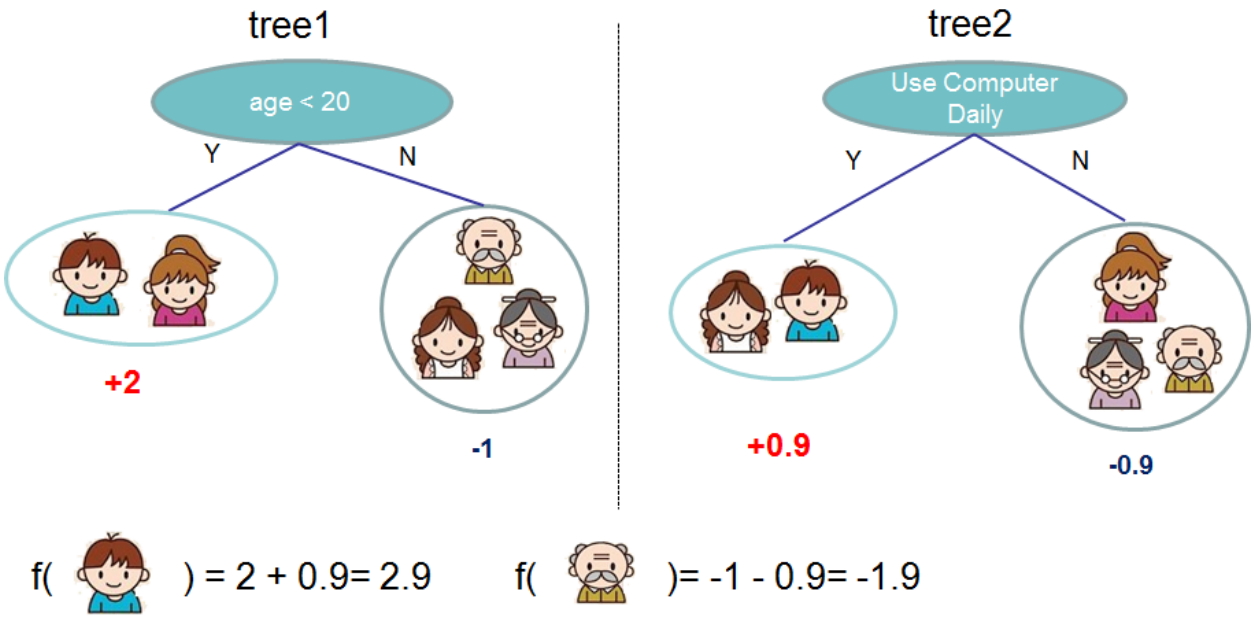


Figure 6.4: An example of a tree ensemble of two trees. Where both trees complement each other. Figure from [10]

6.4 Tree Boosting

To train the model, like all supervised learning models: an objective function with its loss and regularisation must be defined and optimised.

$$Obj(\theta) = \sum_{i}^{n} L(y_i, y_i') + \sum_{k=1}^{K} \Omega(f_k)$$

To find f additive training (Boosting) has to be applied since it is intractable to learn all the trees at once. Therefore add a new function (a new tree) at each time. $y_i'^{(t)}$ is the prediction at each step t:

$$y_i'^{(0)} = 0$$
$$y_i'^{(1)} = f_1(x_i) = y_i'^{(0)} + f_1(x_i)$$
$$y_i'^{(2)} = f_1(x_i) + f_2(x_i) = y_i'^{(1)} + f_2(x_i)$$
$$\dots$$
$$y_i'^{(t)} = \sum_{k=1}^{t} f_k(x_i) = y_i'^{(t-1)} + f_t(x_i)$$

The decision in order to decide which f (tree) is added each time is based on picking the f which optimises our objective function. This is a minimisation problem and further interesting details can be found here [10].

6.5 XGBoost model review

In this section, XGBoost model is described. This method has been implemented with *xgboost* library and all the code is written in python.

As explained above, XGBoost uses a tree ensemble technique where multiple CART trees are used as week learners. The vast majority of parameters are set by default since the implementation provided "is designed to be highly efficient, flexible and portable".

In the table 6.1 the parameters chosen are listed. This attributes have been analytically set.

Description	**Parameter name**	**Value**
Learning rate	eta	0.02
Max tree depth	max_depth	10
Sub-sample ratio of the training instances	sub-sample	0.8
Sub-sample ratio of columns when constructing each tree	colsample_bytree	0.8
Objective function	objective	binary:logistic
Random number seed	seed	0
Evaluation metrics for validation data	eval_metric	auc
Early Stopping		10
Number of trees		1000

Table 6.1: XGBoost chosen parameters

Chapter 7

Convolutional Neural Network

7.1 Neural Network

Neural Networks (NN) or properly referred to as an Artificial Neural Network (ANN). *Dr. Robert Hecht-Nielsen* defines them as: *"... a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs."* [7]. ANNs are computational networks which attempt to simulate the networks of neurons of the biological central nervous system. Is worth mention that this simulation is done in a gross manner.

ANNs allow using very simple operations to solve complex mathematically problems, non-linear problems or stochastic problems. Moreover it has a self degree of self-organising capability that allows it to hold for a wide range of problems.

NNs topology defines what neurons are connected to what others. NNs neurons are typically organised in layers 7.1. This layers are composed of interconnected neurons which contain an activation function. Neurons in a layer are not interconnected in between.

7.2 Neuron

7.2.1 Perceptron

A perceptron works by taking several binary inputs and producing a single binary output, see figure 7.2. A simple rule can be applied to compute the output, defining weights for the inputs. Therefore the output is determined by whether the weighted sum is less or greater than a certain threshold:

$$output = \begin{cases} 0 & if \sum_i w_i x_i \leq threshold \\ 1 & if \sum_i w_i x_i > threshold \end{cases}$$

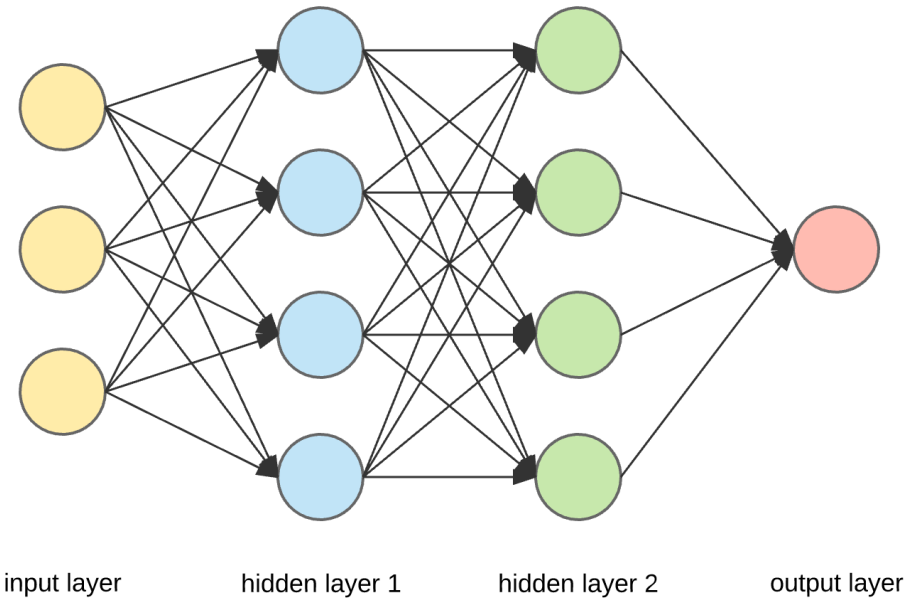


Figure 7.1: An artificial neural network is organised in layers which contain a set of nodes, inspired by a simplification of neurons in a brain. In the figure, each circle represents a neuron and the connections between them represents a connection from the output of one neuron to the input of the following.

For instance, figure 7.1 can be seen as perceptron network where each perceptron take simple decisions that are propagated to the following layer, which in turn decisions are more complex and abstract each time. In this way, a larger network of perceptrons can produce sophisticated decisions.

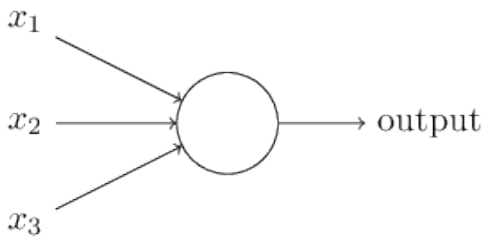


Figure 7.2: A perceptron with 3 inputs x_1, x_2, x_3

7.2.2 Sigmoid neurons

Sigmoid neurons are similar to perceptrons, but differ in that small changes in their weights and bias (threshold) cause only a small change in their output. This small difference results in a neuron capable of learning.

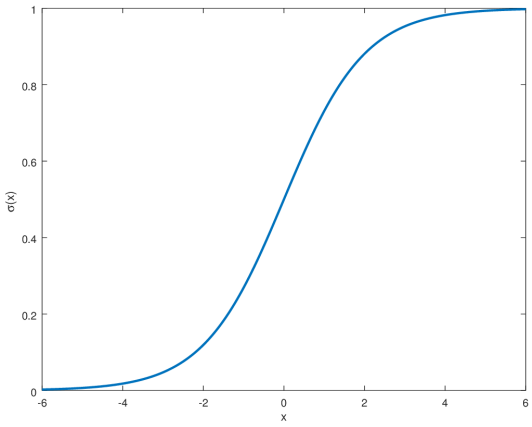


Figure 7.3: The shape of σ function

Like a perceptron 7.2, the sigmoid neuron has inputs whom values are not limited to 0 or 1 instead those are between 0 and 1. The function is called *sigmoid function* σ 7.3:

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

And the output of the neuron is:

$$\frac{1}{1 + e^{(-\sum_i w_i x_i - b)}}$$

7.2.3 Activation function

The activation function $f(\cdot)$, also known as transfer function, defines the state of the neuron itself, i.e is used to determine the output. The concept of neuron can be abstracted from perceptron and sigmoid neuron. The unique thing that differs between them is the activation function. Perceptron transfer function can be seen as an step function, and sigmoid neuron as already explained is seen as a sigmoid function 7.3.

7.3 Learning

Once the neuron type and the topology is given, the behaviour of the network is coded in the weights. Varying weights modifies the action of the network. Learning in a NN is having an algorithm to change the weights.

To modify the weights there is a need to define a Loss function, known as Cost function. In section 6.2 of last chapter, Supervised Learning was explained. Learning means finding

the optimal solution where no other solution has a cost less than the cost of the optimal solution. Learning algorithms try to find the solution with the smallest cost.

7.3.1 Gradient Descent

Optimisation refers to the task of either minimising or maximising some function *f(x)* by altering *x*. In terms of NN, *f(x)* is the loss function 6.2, thus the task is to minimise it.

The *f(x)* derivative (*f'(x)*) tells how to change x in order to make small improvements. Moving x in small steps with the opposite sign of *f'(x)* reduces *f(x)*. This is in fact the gradient descent technique. The goal is to find a minimum in the function where $f'(x) = 0$ and the ideal scenario is to find a global minimum.

7.3.1.1 Stochastic Gradient Descent

Stochastic Gradient Descent (SGD) is an extension of the gradient descent algorithm. As the training set size grows, computing a single gradient step becomes expensive in terms of time.
The name stochastic is because dataset samples are selected randomly instead of as a single group or in the order they appear in the training set.

SGD uses at each step of the algorithm a sample of the training dataset, called minibatch, instead of iteration the entire set. This reduces the amount of time required and outputs very good results in exchange.

7.4 Back-propagation

Back-propagation is an algorithm to compute the gradient of the cost function with respect to the weights of the ANN. Usually the concept back-propagation is misunderstood and thought to be the learning algorithm. Back-propagation refers only to the method for computing the gradient, while stochastic gradient descent, for example, is used to learn.

When an input is fed into a NN, information flows forward through the network to produce an output. This is known as forward propagation. Once it produces an output, a scalar cost is calculated, the algorithm allows the cost value to navigate backwards through the network, in order to computer the gradient. The algorithm does so using a simple cost-less procedure. Information about mathematics and the actual algorithm can be found here [19] and [24].

7.5 Convolutional Neural Networks

Convolutional Neural Networks CNNs are a specific type of neural network for processing data presented in a grid-like topology. Convolutional comes from a mathematical operation called convolution. *Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers.*

7.5.1 Convolution Operation

Convolution is an operation on two functions of a real-valued argument. The convolution operation is typically denoted with an asterisk:

$$s(i,j) = (I * K)(i,j)$$

Where I is referred to as the input, an image for example. K as the kernel and finally *(i,j)* as the axis. Convolutional are often used over more than one axis, see figure 7.4.

$$s(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n)$$

Convolution is commutative.

$$s(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, i-n)K(m,n)$$

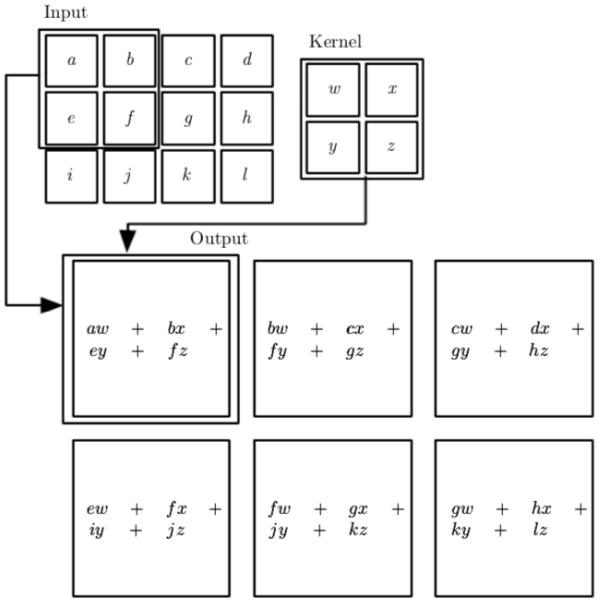


Figure 7.4: An example of 2-D convolution without kernel-flipping. The output is restricted to only positions where the kernel lies within the image. Figure taken from [19].

But this property is not important for a NN implementation, instead cross-correlation is often implemented where the kernel is not flipped:

$$s(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, i+n)K(m,n)$$

7.5.2 Pooling

Typically, a convolutional network layer consists of three stages 7.5. Initially the layer produces a set of linear activations doing several convolutions in parallel, these are run through a nonlinear activation function, such as the rectified linear activation function (ReLu). Finally a pooling function is applied to modify the output of the layer.

Pooling helps to make the representation become approximately invariant to small translations of the input. *Invariance to local translation can be a very useful property if we care more about whether some feature is present than exactly where it is.*

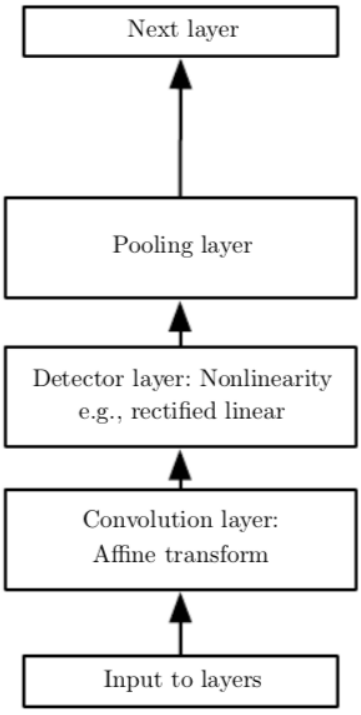


Figure 7.5: Typical convolutional neural network layer. Figure taken from [19].

7.6 CNN model review

This section details the Convolutional Neural Network model used. First the layers are described, finally the various parameters are discussed. The library used is *Keras* [20] and the implementation is in python language.

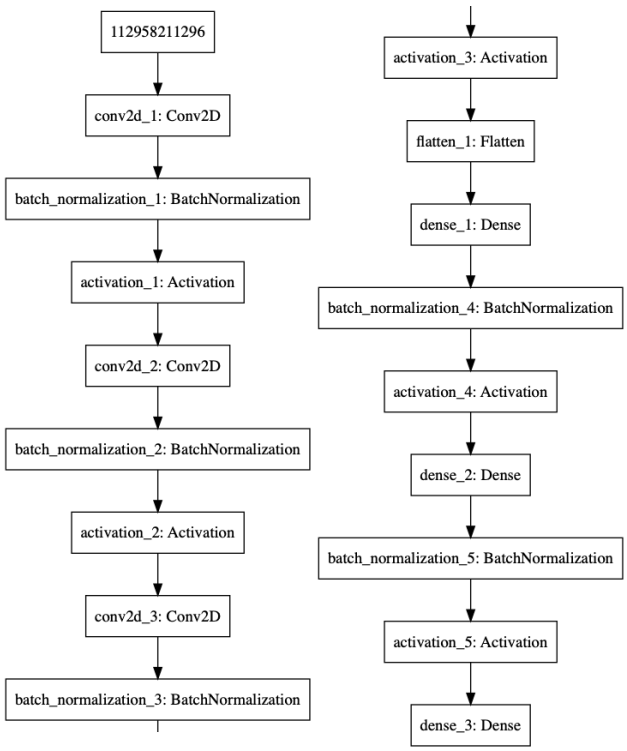


Figure 7.6: Auto-generated image for the CNN model architecture used for classifying 5x5 clusters.

Figure 7.6 is an auto-generated image for the model architecture and details the various layers of the network.

The model for the 5x5 cluster classification layers include:

1. `Convolutional` with `Batch Normalisation` and `Activation` layer. The input is a 2 dimensional matrix of 5 rows and 5 columns size that represents the cluster. The convolution kernel size is 3x3 with a stride of 2. Finally the number of output filters in the convolution is set to 16. Neurons activations are normalised at each batch and the activation function of choice is *ReLu*.

2. `Convolutional` with `Batch Normalisation` and `Activation` Layer. 16 filters of 5 rows and 5 columns size are received. The convolution kernel size is 2x2 with a stride

of 1. The number of output filters in the convolution is set to 8. This layer also normalises the activations at each batch and has *ReLu* as activation function.

3. `Convolutional` with `Batch Normalisation` and `Activation` Layer. 8 filters of 5 rows and 5 columns size are received. The convolution kernel size is 1x1 with a stride of 1. The number of output filters in the convolution is again set to 8. It also applies batch normalisation. and has *ReLu* as activation function.

4. `Flatter` layer. This layer is to fully connect all neurons from now on. Therefore it converts the input size of 8 filters with size 5x5 into 200 neurons.

5. `Dense` with `Batch normalisation` and `Activation` Layer. This layer is a fully connected layer which contains 200 neurons. Neurons activations are normalised at each batch and the activation function of choice is *ReLu*.

6. `Dense` with `Batch normalisation` and `Activation` Layer. This layer is a fully connected layer which contains 20 neurons. It also applies batch normalisation. and has *ReLu* as activation function.

7. `Dense` with `Activation` Layer. This layer contain just 1 neuron and uses a *sigmoid* activation function. It is the output layer of the network and classifies merged π^0 as 0 and γ as 1.

Another model is used for 3x3 cluster sizes classification. The differences are in the convolutional layers, the input sizes are 3 rows and 5 columns. Moreover it only contains 2 Convolutional layers, the first one with a kernel size of 2x2 and the seconds kernel size is 1x1. This are the only differences in the model architecture

The parameters are described in table 7.1. These have been analytically set with a Grid Search algorithm. Where different optimisers where compared: *sgd*, *adam* and *adadelta*. As well as different batch sizes: 5, 10, 25, 50, 100, 150, 200 together with different epochs values: 50, 100, 150, 200.

Batch Size	150
Epochs	50
Metrics	Accuracy
Loss function	Binary cross-entropy
Optimiser	Stochastic gradient descent

Table 7.1: CNN chosen parameters

Chapter 8

π^0/γ classification results

This chapter presents the results obtained with both, the Convolutional Neural Network and eXtreme Gradient Boosting techniques. The CNN model has been explained in section 7. In the other hand, XGBoost model is detailed in section 6. All results presented below have been Cross Validated with a 10 Fold.

First previous work results are replicated and detailed. There are two main previous methods used: use of the *original data* as features, use of *raw digits data* as features.

Subsequent, XGBoost will be used to study the following cases, both for 5x5 and 3x3 Clusters:

- *Raw digits* data
- *Raw cluster* data
- *Raw cluster* and *pre-original* data
- *Raw cluster* and *original* data
- *Raw cluster*, *pre-original* and *original* data

Finally *raw digits* and *raw cluster* data will also be studied with the CNN model both in 5x5 and 3x3 format as input features, treated like an image.

It is important to denote that the three ECAL areas: inner, middle and outer are trained with their specific model and only with data of the specific area, since the cell area is different. This project does not address ECAL regions borders.

8.1 Original and Pre-Original Data

Original data is the current method that LHCb uses to classify π^0 and γ clusters. The original implementation is done with a Neural Network. At the beginning of the project, a replication of the method was done using a BDT, where the results resulted being similar.

Pre-original data was also used to train and test a BDT model, the results were expected to be worse, as this data is a previous step to produce the *original* one.

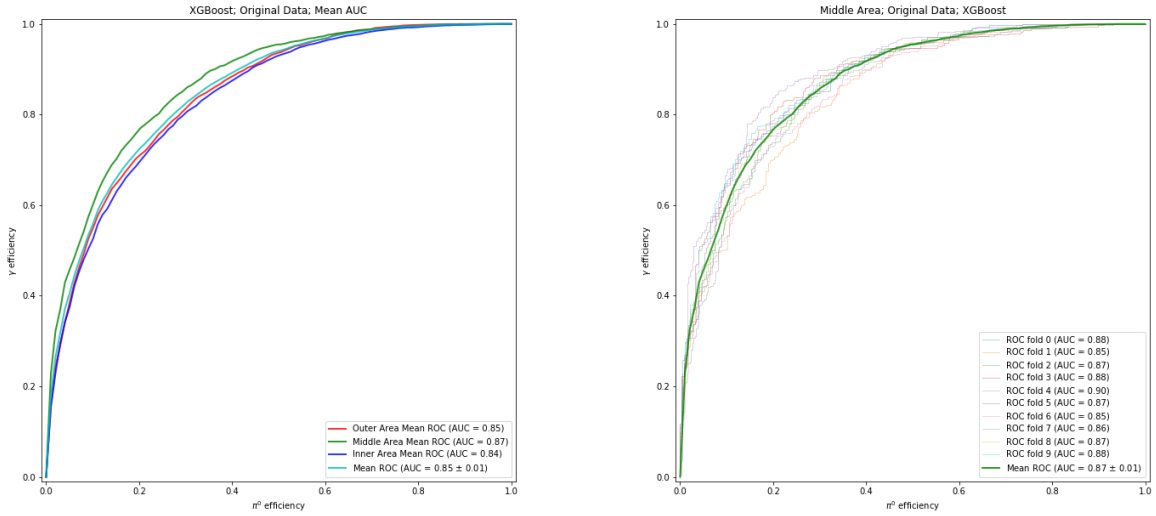
In figure 8.1a the results using *original* data are shown. To do so, ROC curves are employed and the area under the curve (AUC) score is in the plot legend.

In table 8.1, results from both train sources are resumed. As expected *original* data performs way better than *pre-original*, this results confirm that a great job was done in previous studies extracting such features.

Again ROC curves are used in figure 8.1b for the middle area. The mean result is shown together with the cross validation of 10 fold results. Finally in 8.2 the classification performance of the BDT using *original* variables is shown.

ECAL Area	**Original Variables**	**Pre-Original Variables**
Outer	0.85 ± 0.00	0.80 ± 0.01
Middle	0.87 ± 0.00	0.81 ± 0.01
Inner	0.84 ± 0.01	0.75 ± 0.02
Mean	0.85 ± 0.01	0.79 ± 0.03

Table 8.1: Results sing *Original* and *pre-original* data with XGBoost



(a) ROC curves from inner, middle and outer ECAL. In Cian the mean AUC.

(b) ROC curves from middle area In Green the average AUC from the 10 folds.

Figure 8.1: Results from XGBoost trained with *Original* data.

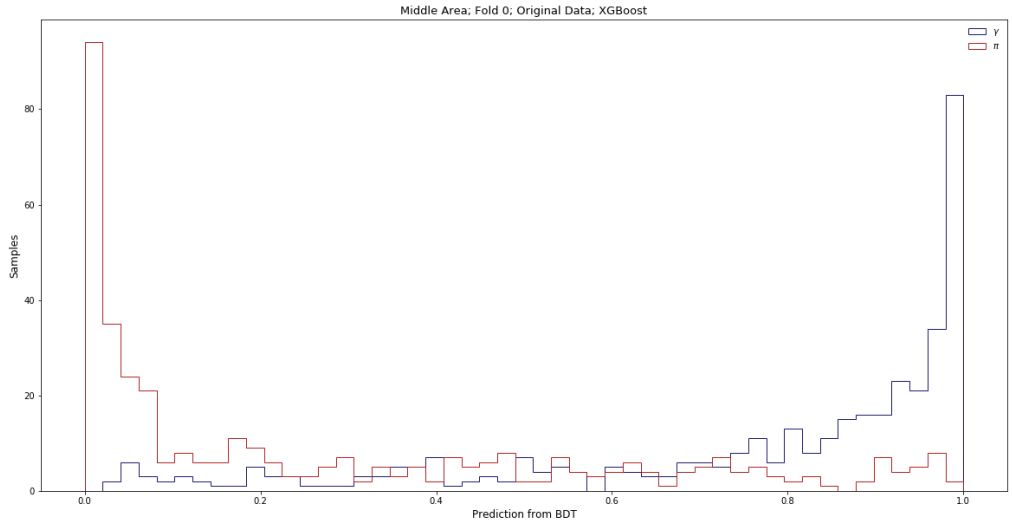


Figure 8.2: XGBoost π^0 and γ classification result trained with *Original* data.

8.2 Raw Data

Scientists from the *University Higher School of Economics* and from the *Yandex School of Data Analysis* presented a poster [5] where important improvements over the *original* method were announced. They used BDT methods together with 5x5 raw energy deposits. This section presents a replication of that implementation as well as a new approaches are proposed. It is important to denote that results differ from their study since this project uses data produced by the upgraded detector description at MC.

The hypothesis of studying the shape of the energy deposits left by particles in the ECAL using raw data was explained in chapter 5. In this section, the product emerging from using *raw digits* and *raw cluster* data is presented.

8.2.1 Raw Digits Data

Leading, the results are replicated. 5x5 primary calorimeter clusters are used and energy at each cell is treated as a feature. Classification performance is attached here 8.3 and ROC curves results can be consulted here A.2. Results from this method are included in table 8.2 as *not processed*.

Consecutive 3x3 size clusters performance is also reviewed on the updated LHCb detector producing more cluster overlaps in the ECAL.

That said, an optimisation is applied where clusters are rotated (explained in section 5.2). Figure 8.4 shows a comparison between processed and not 3x3 and 5x5 primary calorimeter clusters. Complete ROC curves are attached in A.1, A.2, A.3 and A.5.

Lastly, this data has also been used to train a Convolutional Neural Network. Using this approach, the clusters are treated as images where each cell is a pixel. The results from this method do not meet the baseline set by XGBoost. But interesting information can be taken from it, see figures A.6 and A.4 for a per area detailed comparison between 3x3 and 5x5 clusters performance using CNN.

Table 8.2 sums up all the results obtained using this *raw digits* data. The middle area has the highest score with all the methods and the processed data has a slightly better performance. In the other hand, CNN are always below XGBoost results.

Cluster size	ECAL area	XGBoost		CNN	
		Not processed	Processed	Not processed	Processed
3x3	Outer	0.89 ± 0.01	0.89 ± 0.01	0.83 ± 0.02	0.84 ± 0.03
	Middle	0.89 ± 0.01	0.90 ± 0.01	0.86 ± 0.02	0.88 ± 0.02
	Inner	0.88 ± 0.00	0.89 ± 0.01	0.86 ± 0.01	0.87 ± 0.01
	Mean	0.89 ± 0.00	0.90 ± 0.00	0.85 ± 0.01	0.86 ± 0.01
5x5	Outer	0.91 ± 0.01	0.91 ± 0.01	0.83 ± 0.02	0.85 ± 0.01
	Middle	0.91 ± 0.01	0.92 ± 0.01	0.85 ± 0.02	0.88 ± 0.01
	Inner	0.90 ± 0.00	0.91 ± 0.00	0.86 ± 0.01	0.88 ± 0.01
	Mean	0.91 ± 0.01	0.91 ± 0.00	0.85 ± 0.01	0.87 ± 0.01

Table 8.2: Comparison between 3x3 and 5x5 clusters using XGBoost and CNN methods trained with *raw digits* data.

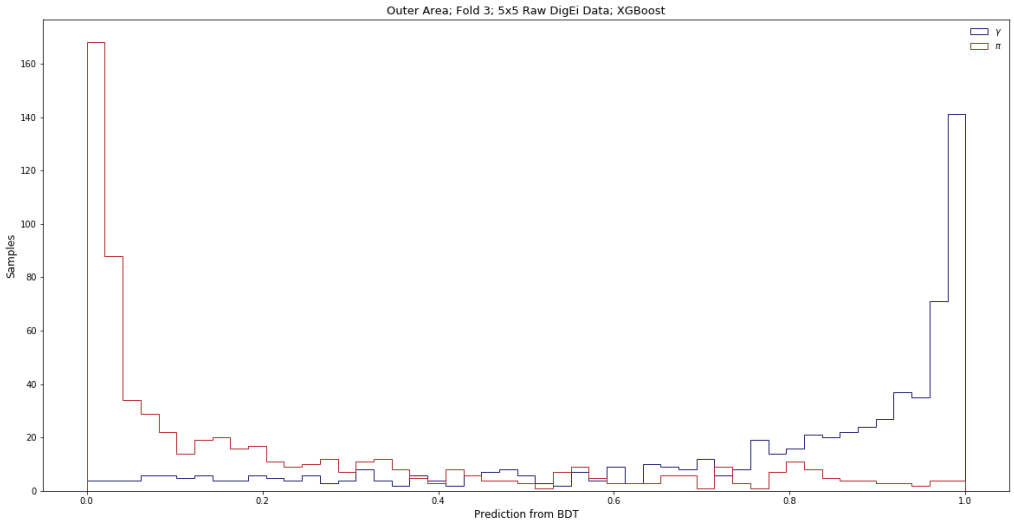
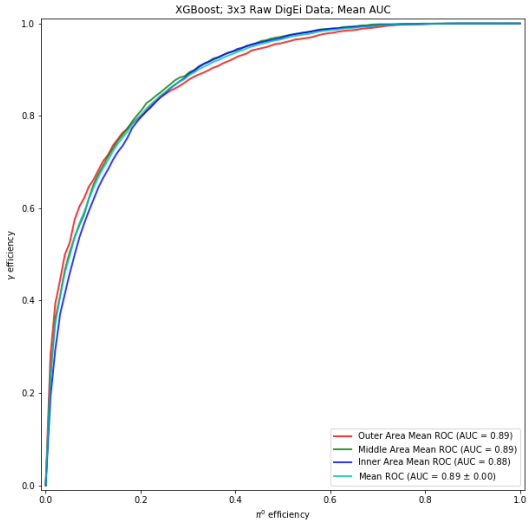
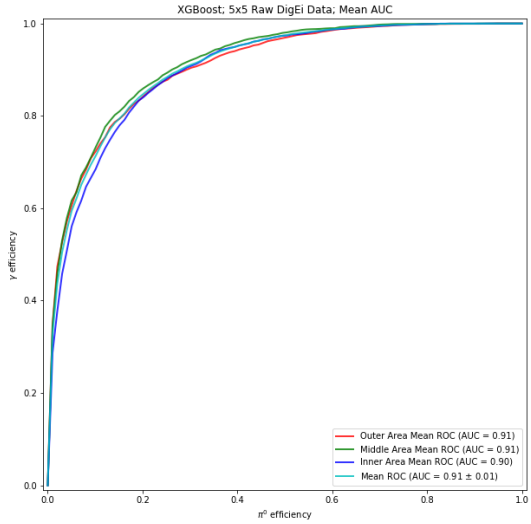


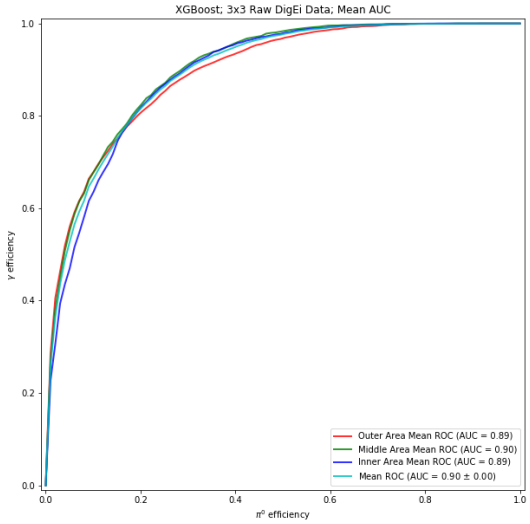
Figure 8.3: Neutral particle classification using unprocessed raw digits data in outer region with XGBoost model.



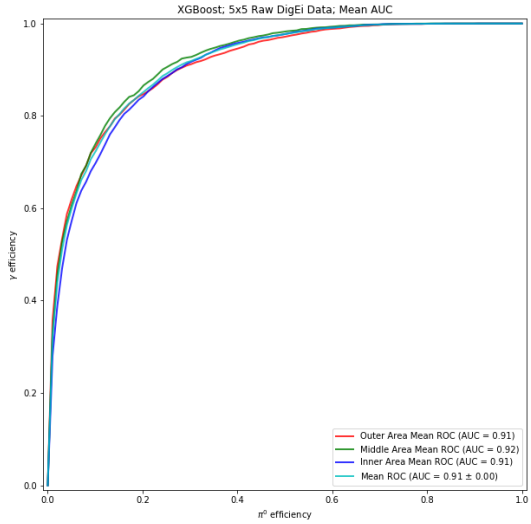
(a) 3x3 unprocessed clusters.



(b) 5x5 unprocessed clusters.



(c) 3x3 processed clusters.



(d) 5x5 processed clusters.

Figure 8.4: ROC curves comparison obtained with the replicated method and same method with optimised *raw digits* format using XGBoost technique.

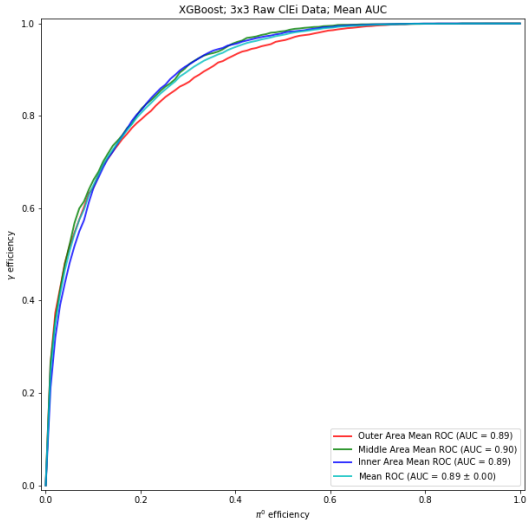
8.2.2 Raw Cluster Data

Due to the overlapping presented in primary clusters straight from the detector, the same methods are applied to reconstructed clusters using a cellular automaton, in chapter 5 comparison between *raw digits* and *raw cluster* is detailed. The reconstructed clusters size studied are 3x3 and 5x5 cells extension and the Machine Learning techniques used are XGBoost and CNN. Here only processed clusters are used, since the results are better.

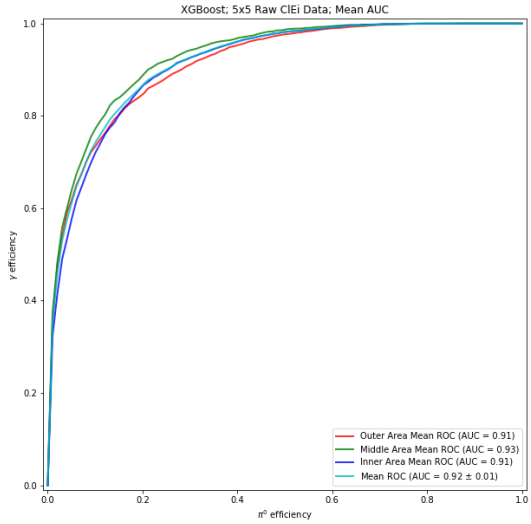
Table 8.3 summarises the AUC score results for each ML method. Note that XGBoost results are even better than processed *raw digits* clusters and again the middle is in general the outstanding region. CNN results are quite similar and still below BDT ones. Figure 8.5 gathers all the ROC curves for each method.

Cluster size	**ECAL area**	**XGBoost**	**CNN**
3x3	**Outer**	0.89 ± 0.01	0.84 ± 0.03
	Middle	0.90 ± 0.01	0.88 ± 0.01
	Inner	0.89 ± 0.00	0.87 ± 0.02
	Mean	0.89 ± 0.00	0.86 ± 0.02
5x5	**Outer**	0.91 ± 0.01	0.84 ± 0.02
	Middle	0.93 ± 0.01	0.87 ± 0.02
	Inner	0.91 ± 0.00	0.86 ± 0.02
	Mean	0.92 ± 0.01	0.87 ± 0.01

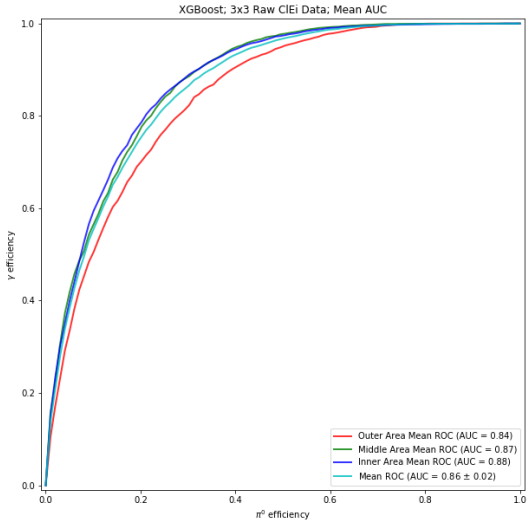
Table 8.3: Comparison between 3x3 and 5x5 clusters using XGBoost and CNN methods trained with *raw clusters* data.



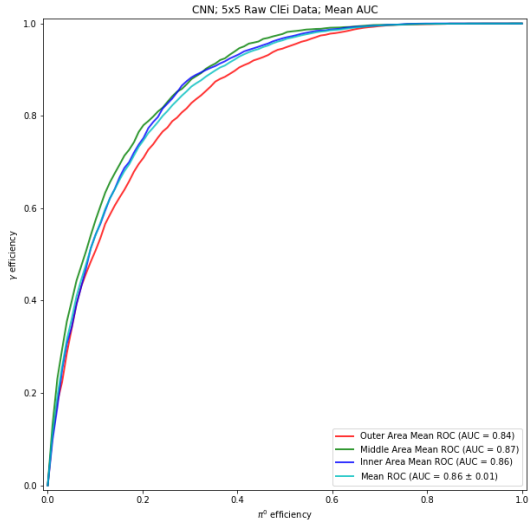
(a) Using XGBoost and 3x3 clusters.



(b) Using XGBoost and 5x5 clusters.



(c) Using CNN and 3x3 clusters.



(d) Using CNN and 5x5 clusters.

Figure 8.5: ROC curves comparison obtained with *raw clusters* data using XGBoost and CNN technique.

8.3 Combining Raw Data and Original Data

Finally, the last tests are possible combinations of features from each type of data. The main purpose is to improve the results further, moreover to find out which datasets are "equivalent" i.e, since all variables map the same data the objective is to find out the correlation between them.

Accordingly, the below combinations of features to train a BDT are studied:

- Raw digits and original data.
- Raw clusters and original data.
- Raw clusters and pre-original data.
- Raw clusters, original and pre-original data.

The following table 8.4 summarises all the scores outcomes.

Cluster size	**ECAL area**	**Raw digits and Original**	**Raw cluster and Original**	**Raw cluster and pre-original**	**Raw cluster, original and pre-original**
3x3	**Outer**	0.90 ± 0.01	0.90 ± 0.01	0.89 ± 0.01	0.90 ± 0.01
	Middle	0.92 ± 0.01	0.92 ± 0.01	0.91 ± 0.01	0.92 ± 0.01
	Inner	0.90 ± 0.01	0.90 ± 0.01	0.90 ± 0.00	0.90 ± 0.01
	Mean	0.91 ± 0.01	0.91 ± 0.01	0.90 ± 0.01	0.91 ± 0.01
5x5	**Outer**	0.92 ± 0.01	0.92 ± 0.01	0.91 ± 0.01	0.91 ± 0.01
	Middle	0.93 ± 0.01	0.93 ± 0.01	0.93 ± 0.01	0.93 ± 0.01
	Inner	0.91 ± 0.00	0.91 ± 0.00	0.91 ± 0.00	0.91 ± 0.00
	Mean	0.92 ± 0.01	0.92 ± 0.01	0.92 ± 0.01	0.92 ± 0.01

Table 8.4: Comparison between 3x3 and 5x5 clusters using XGBoost method trained with different combinations of data-groups.

In essence, *original* and *pre-original variables* aggregate similar features when combined with raw data. Significantly improvements in 3x3 cluster size can be pointed out. Concluding that extra features are added when any of the *pre-original* or *original variables* are used

alongside raw data. Combining these methods results in the best classification performance so far, specially for *raw data* together with *original varibles*.

8.4 Comparison

Heading, the best results using XGBoost method is recapped in the tables 8.5 and 8.6 included below.

Cluster size	**3x3**			
ECAL area	**Outer**	**Middle**	**Inner**	**Mean**
Raw digits	0.89± 0.01	0.90 ± 0.01	0.89 ± 0.00	0.90 ± 0.00
Raw cluster	0.89 ± 0.01	0.90 ± 0.01	0.89 ± 0.00	0.89 ± 0.00
Raw digits and Original	0.90 ± 0.01	0.92 ± 0.01	0.90 ± 0.01	0.91 ± 0.01
Raw cluster and Original	0.90 ± 0.01	0.92 ± 0.01	0.90 ± 0.01	0.91 ± 0.01

Table 8.5: Best results obtained with a 3x3 cluster size using XGBoost.

Either raw digits and raw clusters together with original variables obtain the best performance in 3x3 cluster size.

Moving on 5x5 size clusters the best results are again when raw data is combined with original variables, nevertheless, raw clusters by their own perform near identically.

Finally table 8.7 summarises the different models together with the original baseline. Although the best results are obtained with a XGBoost method, the CNN technique does also improve the baseline results.

Cluster size	**5x5**			
ECAL area	**Outer**	**Middle**	**Inner**	**Mean**
Raw digits	0.91 ± 0.01	0.92 ± 0.01	0.91 ± 0.00	0.91 ± 0.00
Raw cluster	0.91 ± 0.01	0.93 ± 0.01	0.91 ± 0.00	0.92 ± 0.01
Raw digits and Original	0.92 ± 0.01	0.93 ± 0.01	0.91 ± 0.00	0.92 ± 0.01
Raw cluster and Original	0.92 ± 0.01	0.93 ± 0.01	0.91 ± 0.00	0.92 ± 0.01

Table 8.6: Best results obtained with a 5x5 cluster size using XGBoost

ECAL area	**Outer**	**Middle**	**Inner**	**Mean**
Original data XGBoost	0.85 ± 0.00	0.87 ± 0.00	0.84 ± 0.01	0.85 ± 0.01
5x5 Raw data CNN	0.85 ± 0.01	0.88 ± 0.01	0.88 ± 0.01	0.87 ± 0.01
5x5 Raw and original XGBoost	0.92 ± 0.01	0.93 ± 0.01	0.91 ± 0.00	0.92 ± 0.01

Table 8.7: Best results obtained with CNN and XGBoost together with the baseline result.

Ending this chapter, the distribution of the output of the classifier obtained with the ML algorithms combined with the different data-sources present in table 8.7 is shown in figures 8.6 for XGBoost trained with original data, 8.7 for CNN trained with *raw digits* data and 8.8 for XGBoost trained with *raw cluster* and *original* data. Moreover, figure 8.9, gathers all the ROC curves for the last and the best result obtained in this study.

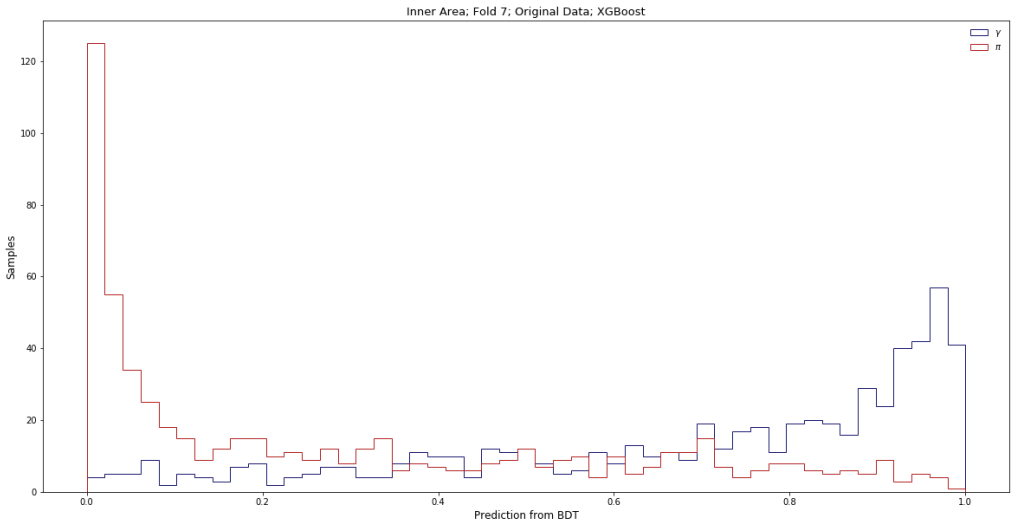


Figure 8.6: Class separation resulting from XGBoost method using *original* data.

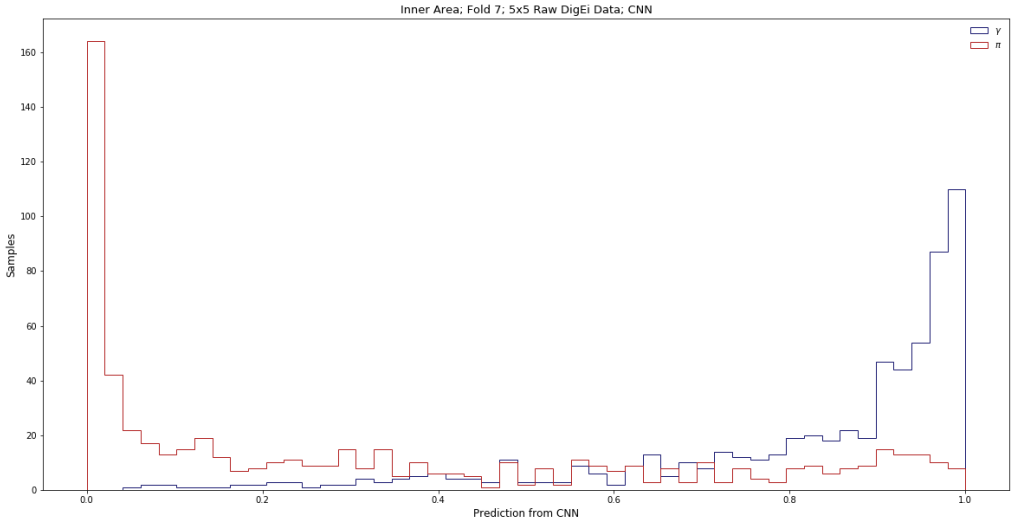


Figure 8.7: Class separation resulting from CNN method using *raw digits* data

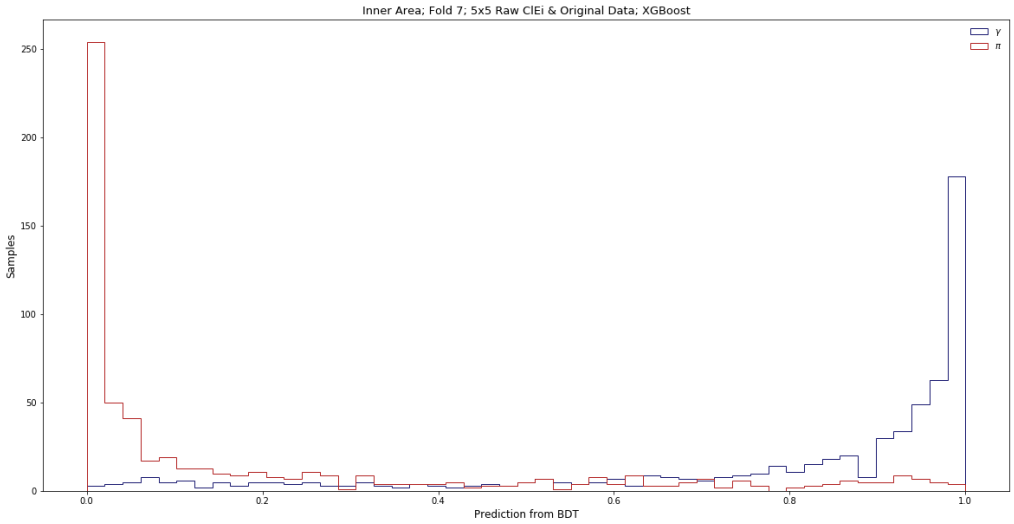
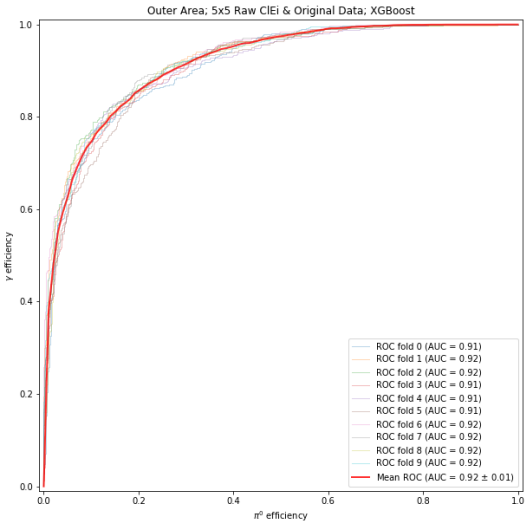
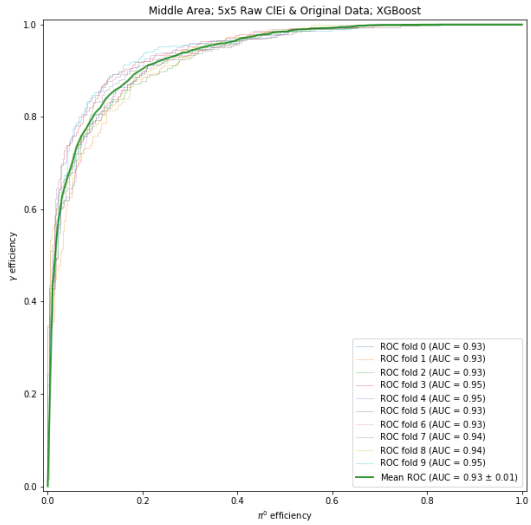


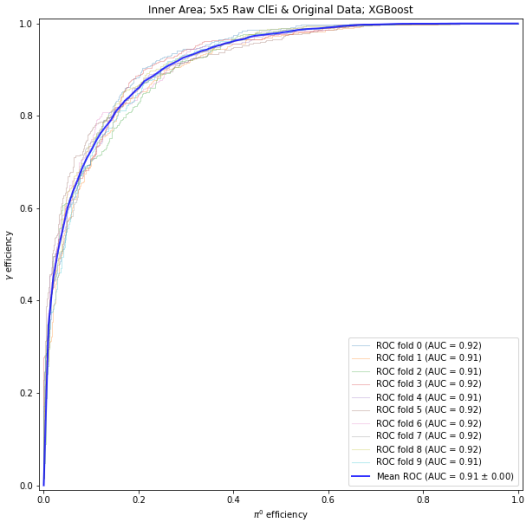
Figure 8.8: Class separation resulting from XGBoost method using *raw cluster* and *original* data



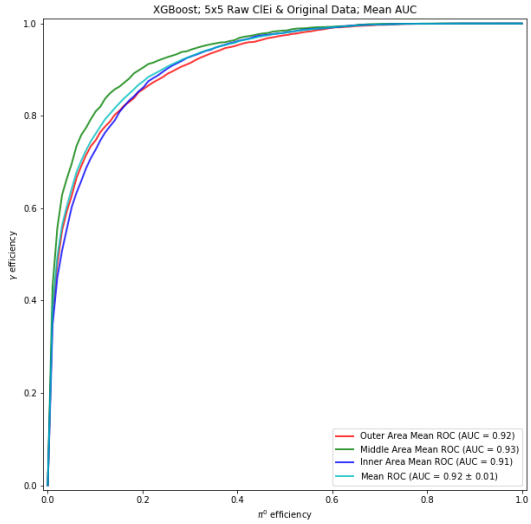
(a) ROC curves for the outer region.



(b) ROC curves for the middle region.



(c) ROC curves for the inner region.



(d) ROC curves comparison and average.

Figure 8.9: Best auc scores obtained. *Raw clusters* and *original* data using XGBoost technique.

Chapter 9

Conclusions

In order to conclude the results described, different approaches have been studied at Monte Carlo data level.

Before starting the implementation, I had to study the CERN institution, the LHC and more specific the ECAL detector in the LHCb. Although knowing beforehand CERN and some achievements, I realised the magnitude of the project, how it's structured, the different detectors found, the simulations... More important, I found out the work behind thousands of ideas each sub-detector is surrounded. Reading through lots of articles I was stunned the amount of Nobel prices obtained through designs present at CERN.

Once the problem was understood, research of the Machine Learning algorithms began. I had to remember the concepts behind decision trees and neural networks. Later a master understanding of the libraries capabilities had to be settled. Both algorithms have a well known reputation and enormous potential has been proved in multiple fields.
Decision trees have unbelievable power, during my degree I wasn't able to see the capacity of such algorithms, now I really have in mind what are they capable off. Learning about it, multiple examples were found. XGBoost is a winner algorithm used in many competitions due its flexible capacity.
In the other hand I had great experience with neural networks during my degree. Reading multiple references about CNN I realised the proved reputation they had dealing with images.
Following, the replication of previous tools with 3x3 cluster size began. Started with 3x3 cluster sizes because the increased luminosity in the upgraded detector produces overlapping between clusters, hence, grabbing smaller size clusters seamed to be the key.
The original variables and raw clusters previous tools were successfully replicated.
Later on, the CNN implementation commenced. The results were bad and lots of tests through the network architecture was done together with Grid Searches to optimally tune the network parameters.
By that time, results obtained with XGBoost were great, the original variables together

with raw clusters were already tested and the results outperformed any approach tested and replicated until the date. Through the experimentation of modifying the CNN model architecture and inspecting the results, clues of using bigger clusters arose.

Therefore 5x5 cluster data was generated at Monte Carlo simulation. The same ML algorithms with minor changes where applied. The outputs were better and together the optimisation done regarding raw clusters rotation resulted in a successful product.

Although the CNN results in this project weren't as good as one could expect at the beginning and given the best results obtained with any method are always in the middle region of the ECAL, the conclusions brings the discussion to further studies regarding tests with the same methods with an ECAL cell description smaller in every area. Hence the outer and inner areas results are always similar, the conclusion is that inner cells which have greater resolution that any other region also have the greatest amount of overlapping. In the other hand outer cells do not suffer heavy overlapping but the resolution is very the poorest. The middle region has the better from both regions, less overlapping than inner area and more resolution than the outer resulting in a key combination. More resolution in every area could lead to higher accuracy respecting the particle classification.

Finally I would like to end this project highlighting the great opportunity and experience I had with the research group *DS4DS*, since it has been my first approach to a truly research project together with working for an important institution like CERN.

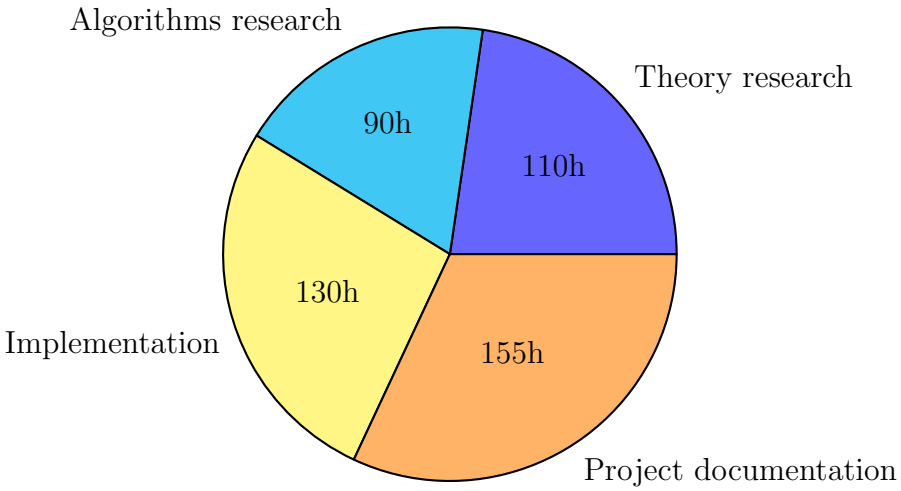


Figure 9.1: Distribution of time and amount of hours spent per tasks.

Appendix A

XGBoost and CNN results

A.0.1 Unprocessed 3x3 Raw Digits Data

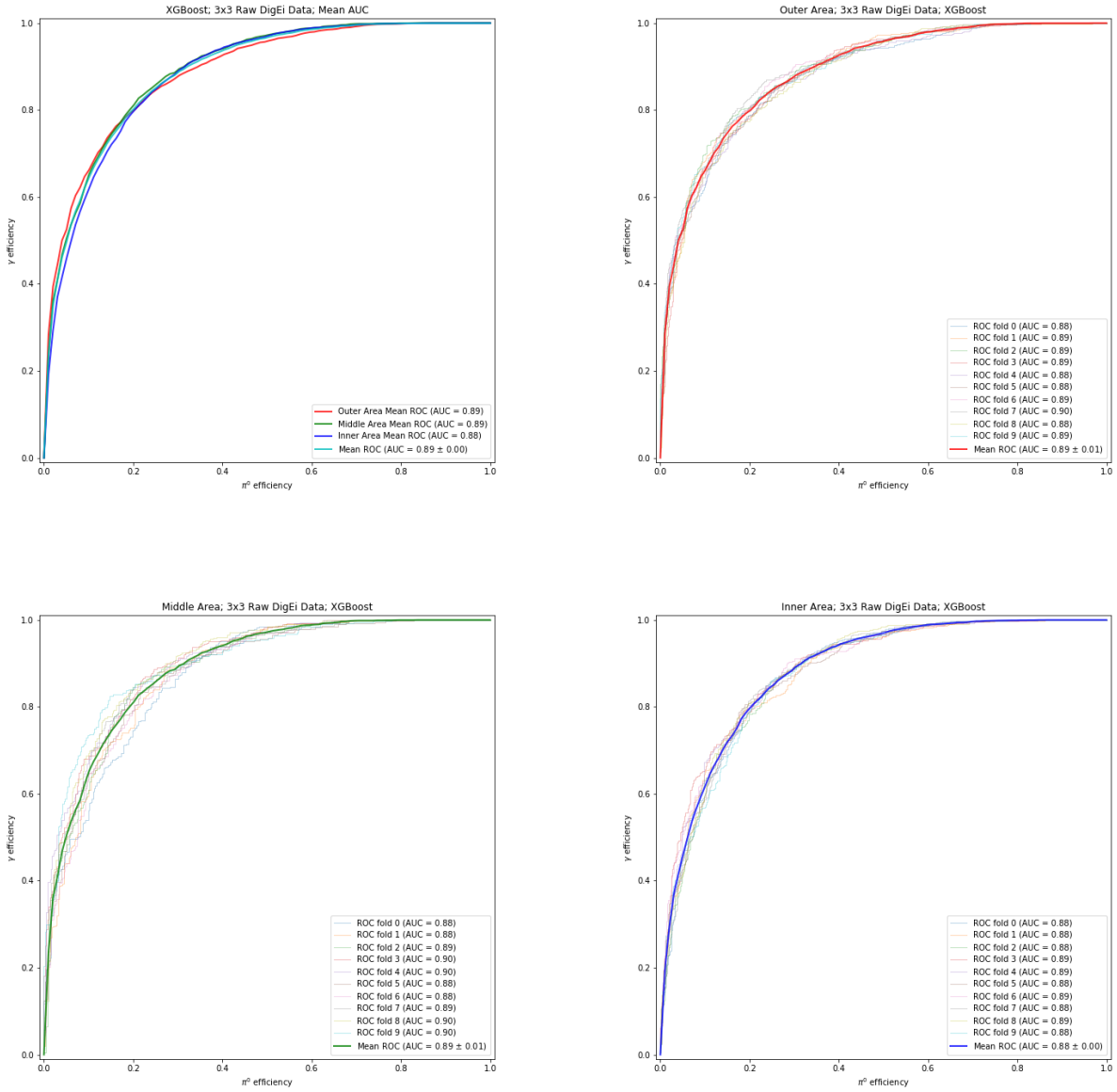


Figure A.1: XGBoost results using unprocessed 3x3 Raw Digits Data.

A.0.2 Unprocessed 5x5 Raw Digits Data

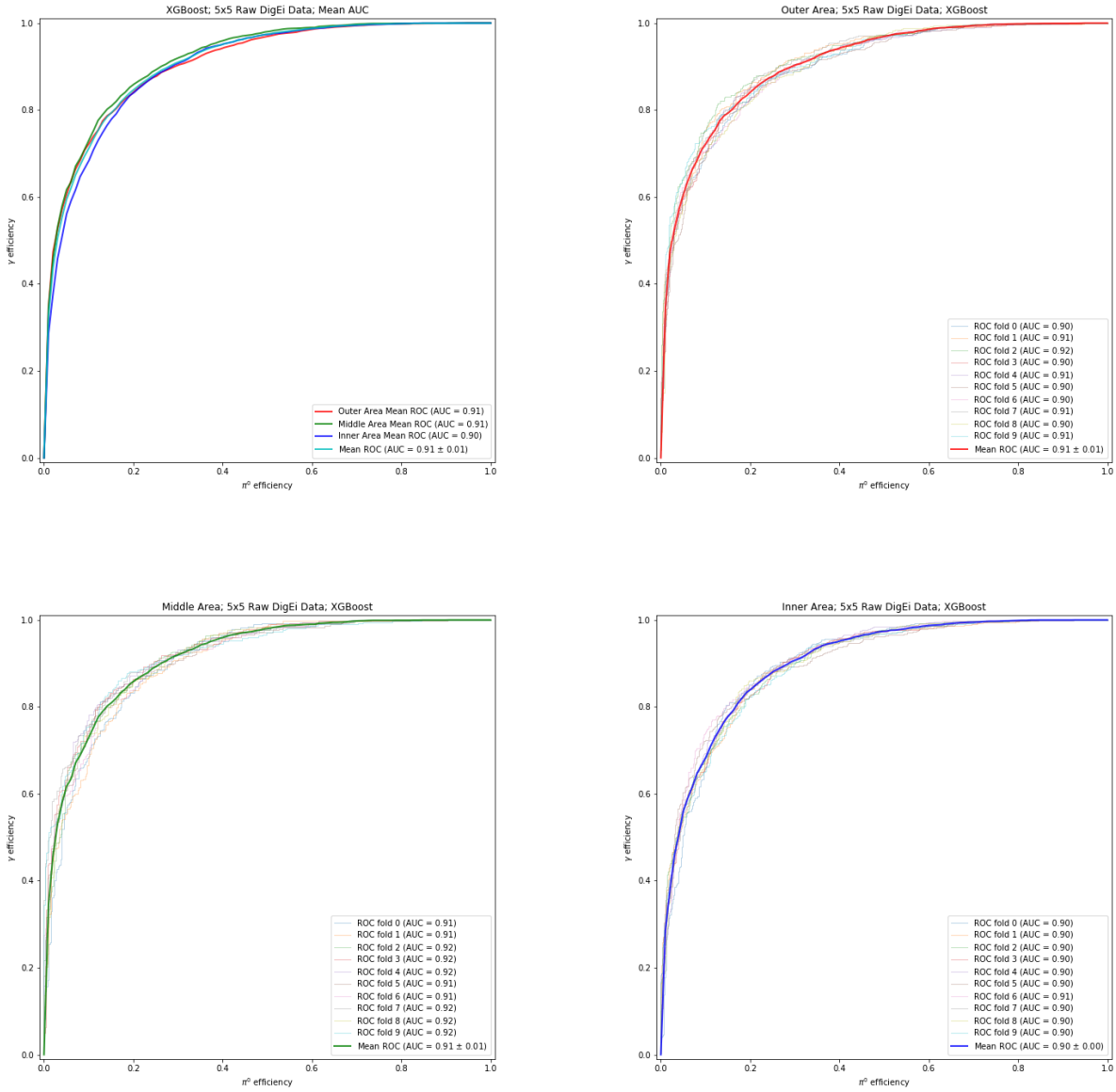


Figure A.2: XGBoost results using unprocessed 5x5 Raw Digits Data.

A.0.3 3x3 Raw Digits Data

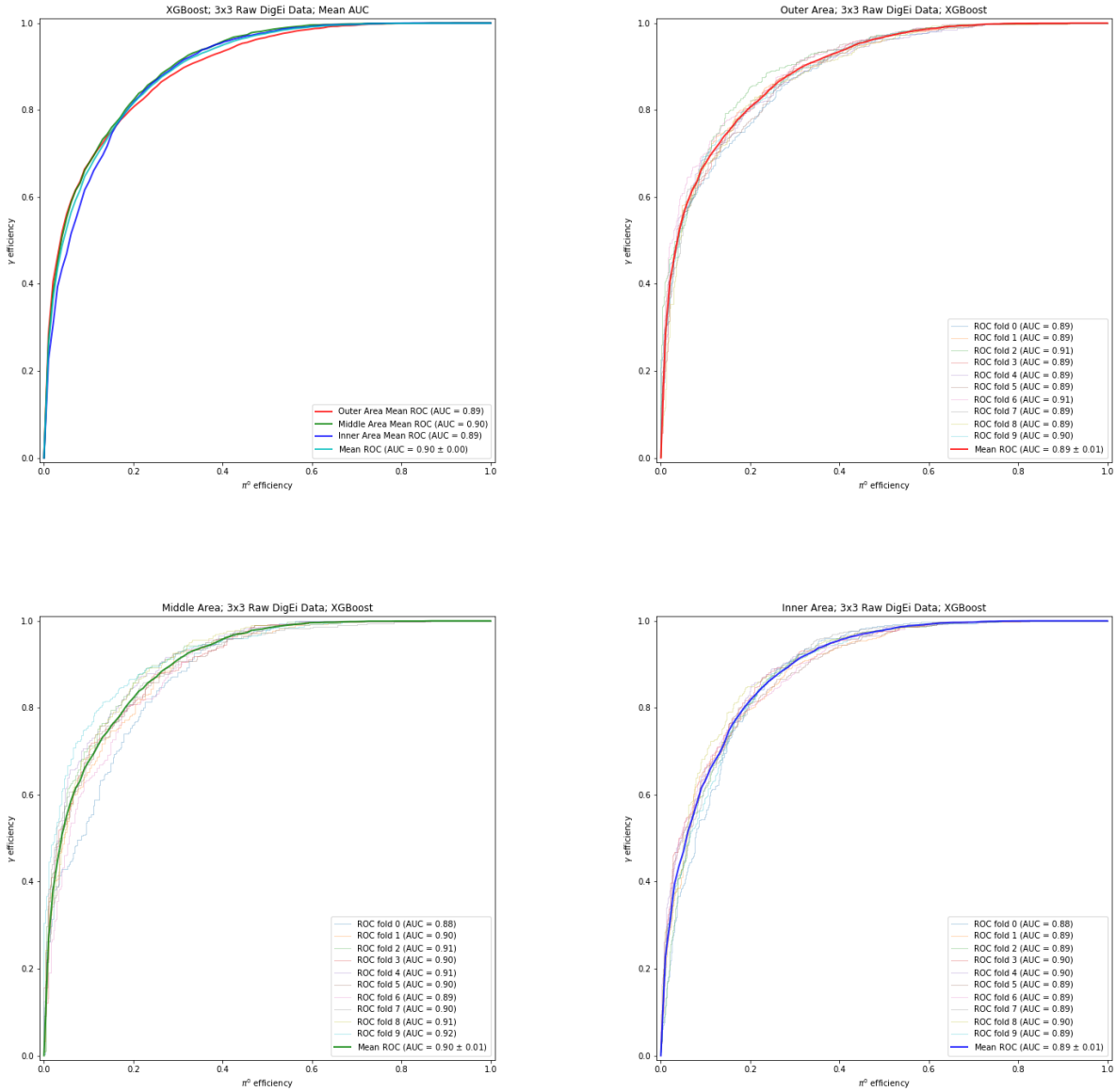


Figure A.3: XGBoost results using 3x3 Raw Digits Data.

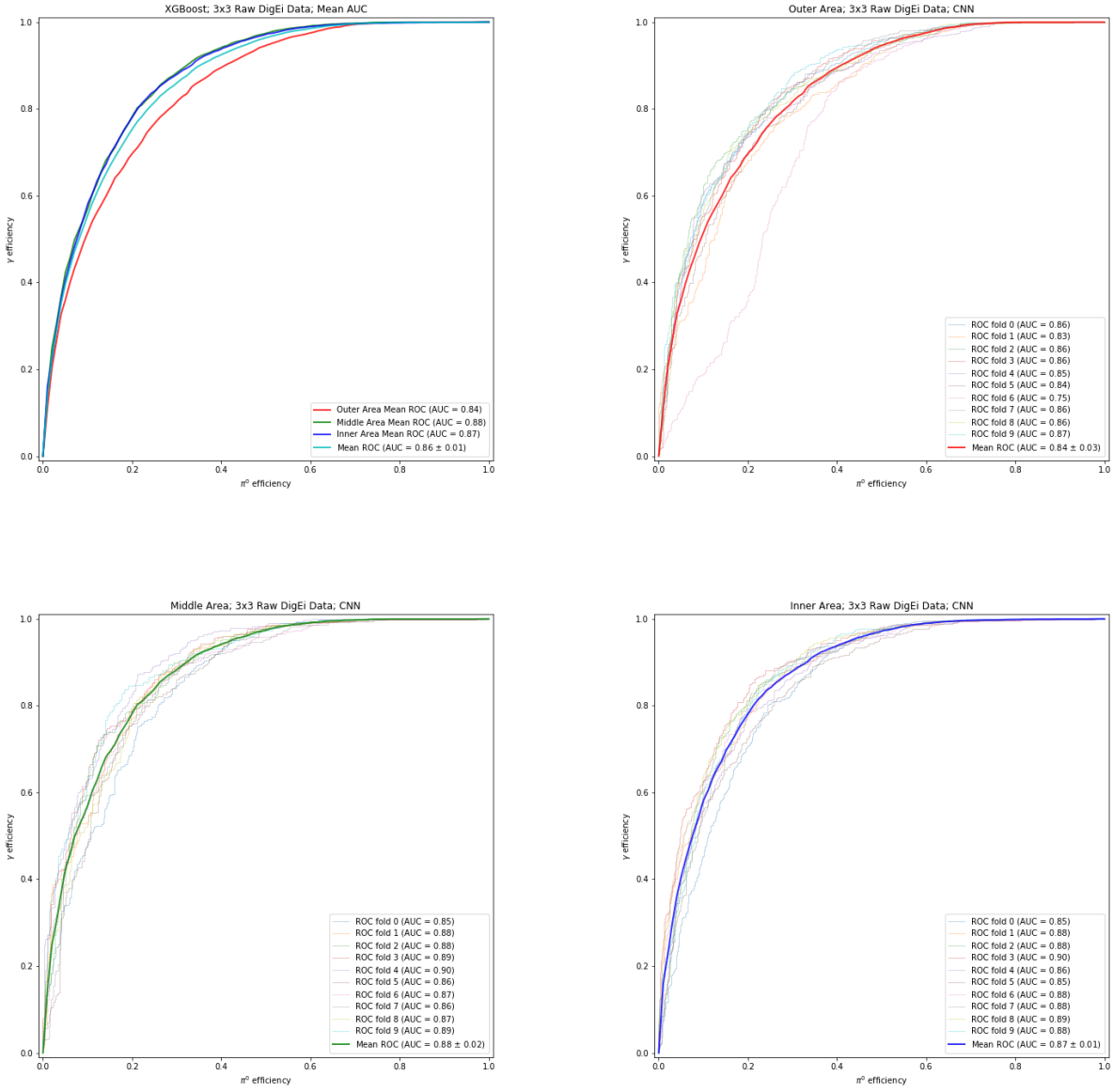


Figure A.4: CNN results using 3x3 Raw Digits Data.

A.0.4 5x5 Raw Digits Data

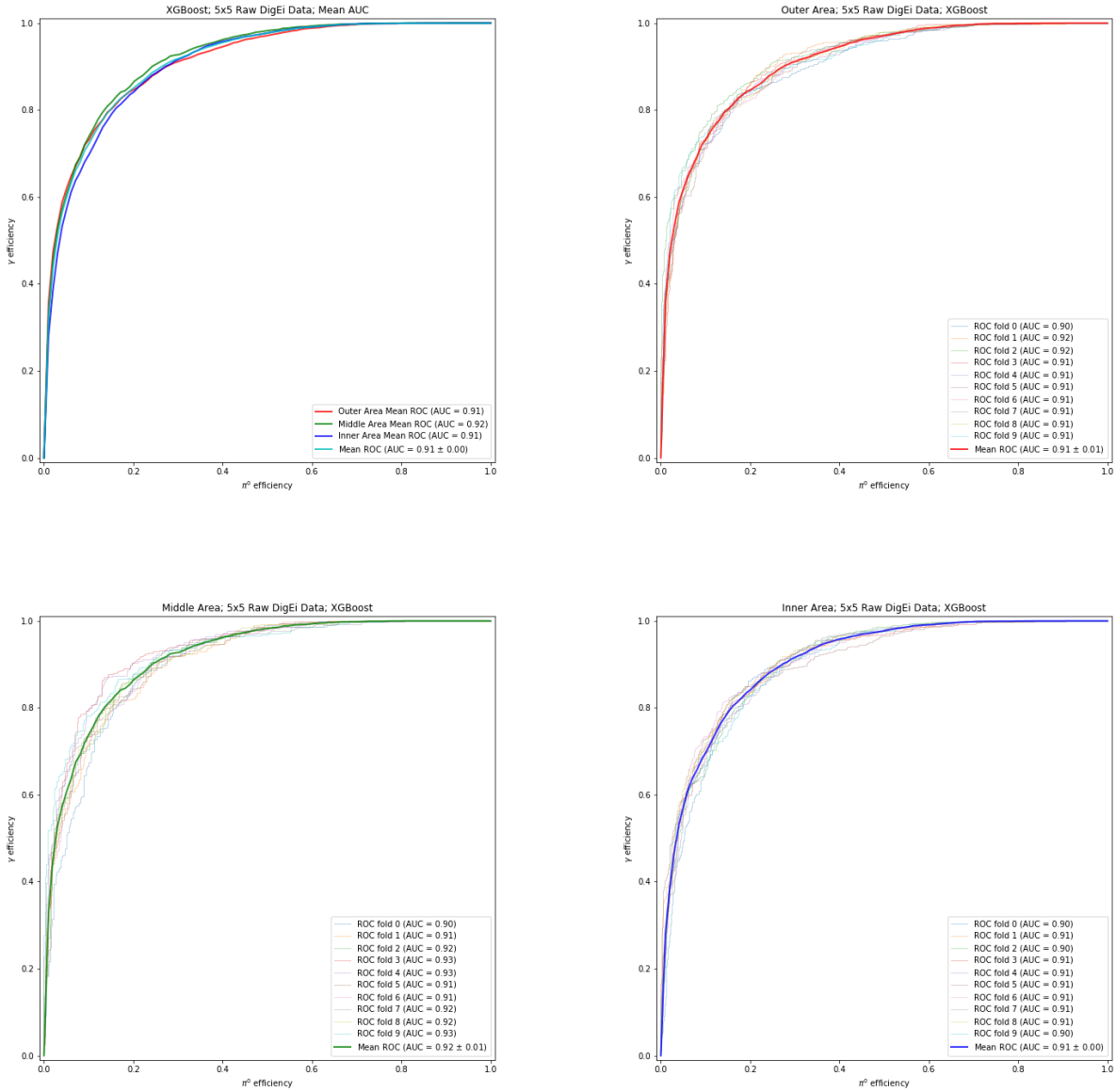


Figure A.5: XGBoost results using 5x5 Raw Digits Data.

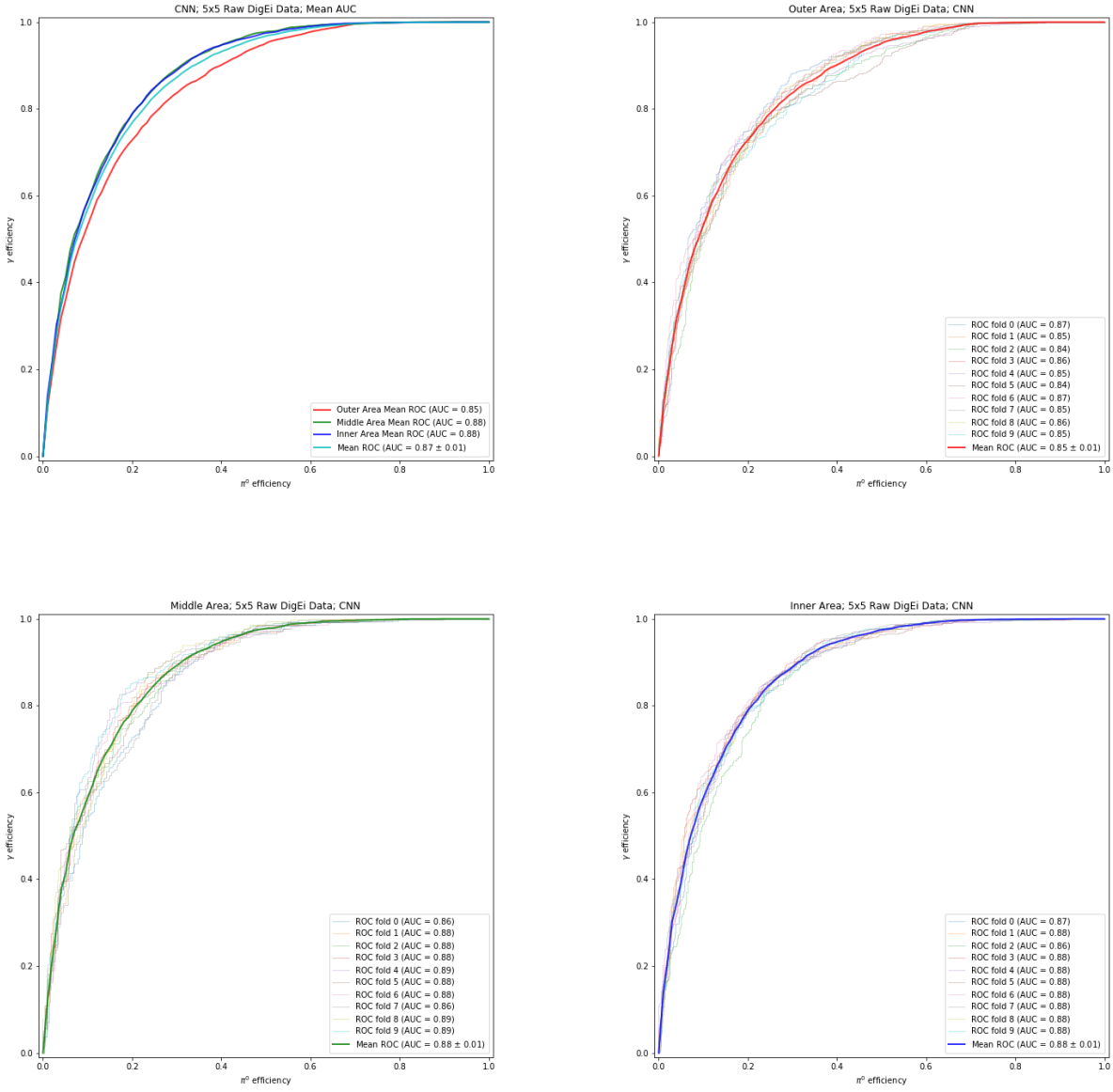


Figure A.6: CNN results using 5x5 Raw Digits Data.

A.0.5 3x3 Raw Clusters Data

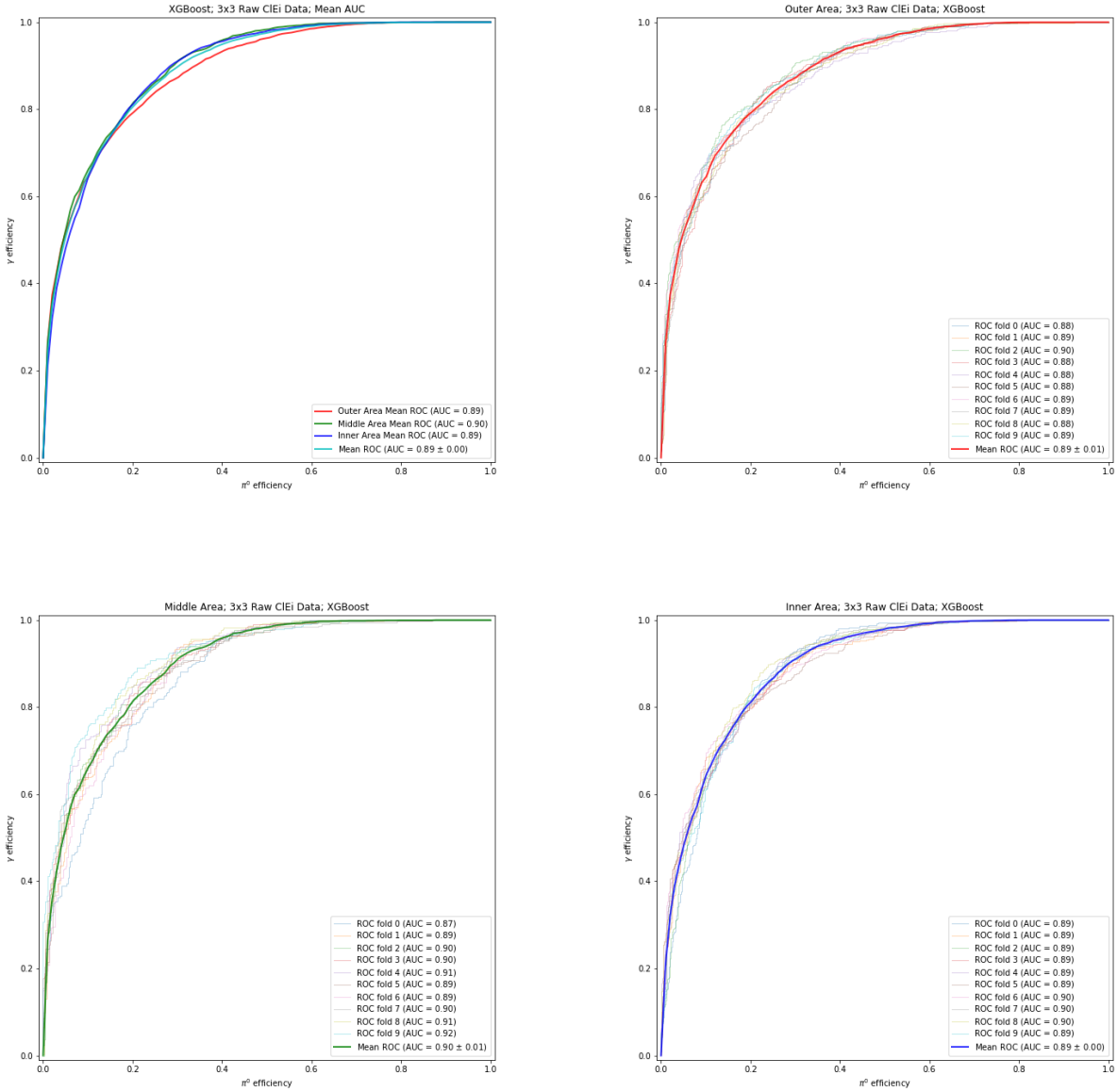


Figure A.7: XGBoost results using 3x3 Raw Clusters Data.

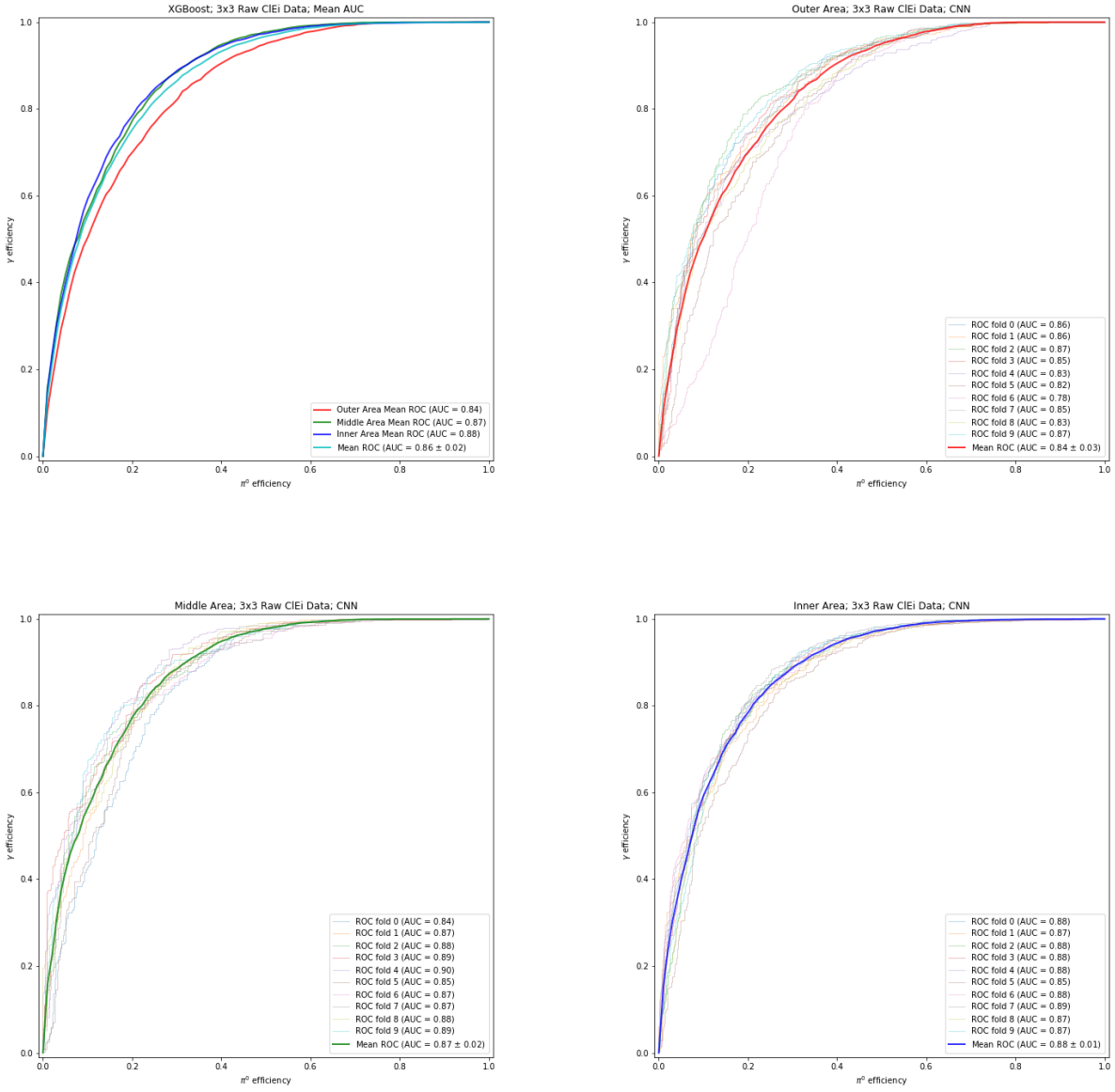


Figure A.8: CNN results using 3x3 Raw Clusters Data.

A.0.6 5x5 Raw Clusters Data

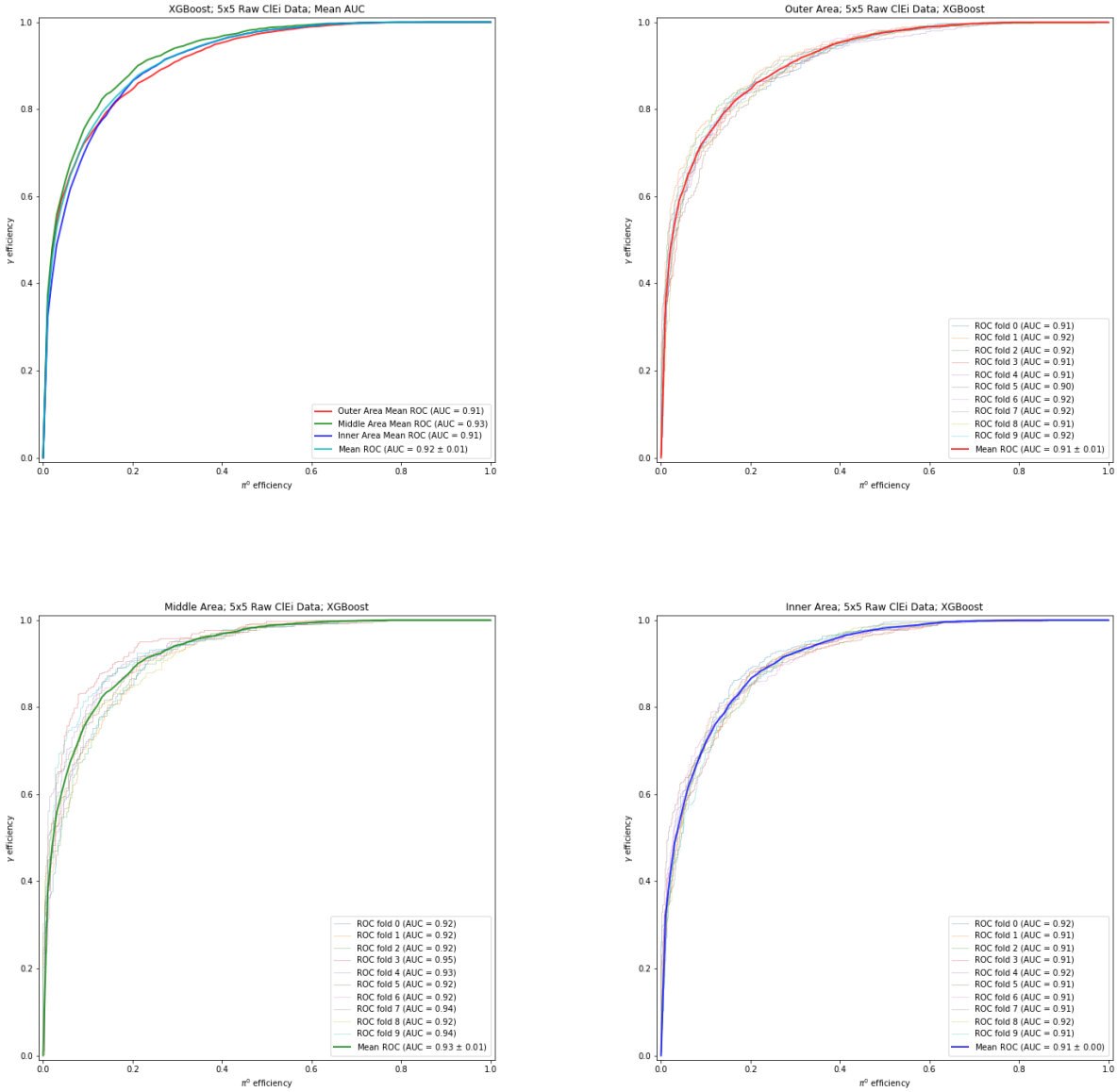


Figure A.9: XGBoost results using 5x5 Raw Clusters Data.

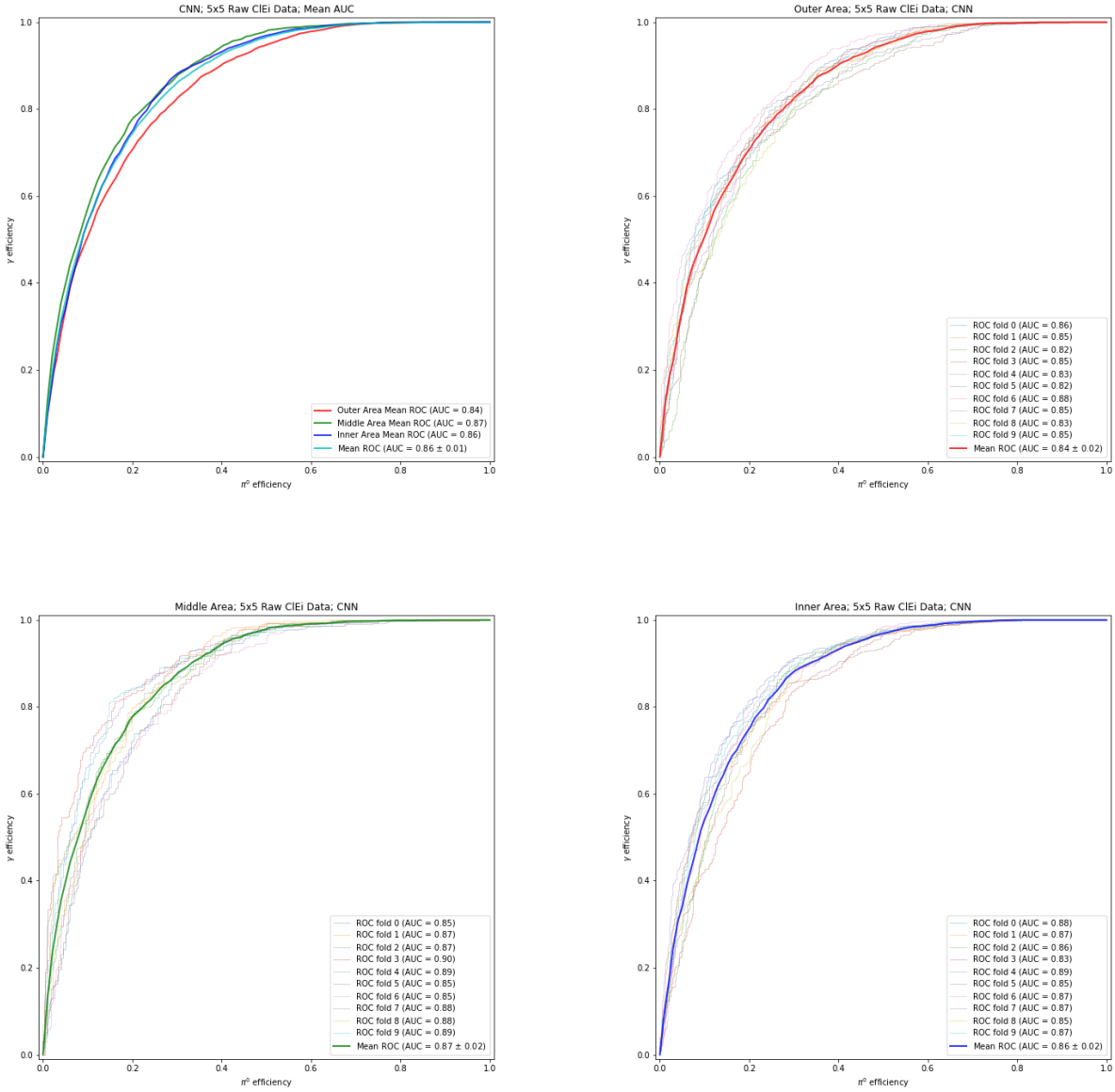


Figure A.10: CNN results using 5x5 Raw Clusters Data.

Bibliography

[1] G. Aad et al. "The ATLAS Experiment at the CERN Large Hadron Collider". In: *JINST* 3 (2008), S08003. DOI: 10.1088/1748-0221/3/08/S08003.

[2] K. Aamodt et al. "The ALICE experiment at the CERN LHC". In: *JINST* 3 (2008), S08002. DOI: 10.1088/1748-0221/3/08/S08002.

[3] A. Augusto Alves Jr. et al. "The LHCb Detector at the LHC". In: *JINST* 3 (2008), S08005. DOI: 10.1088/1748-0221/3/08/S08005.

[4] A Aref'ev et al. *Design, construction, quality control and performance study with cosmic rays of modules for the LHCb electromagnetic calorimeter*. Tech. rep. LHCb-2007-148. CERN-LHCb-2007-148. Geneva: CERN, Jan. 2008. URL: http://cds.cern.ch/record/1080559.

[5] Alexey Boldyrev. "Machine Learning approach to boosting neutral particles identification in the LHCb calorimeter". Mar. 2019. URL: https://cds.cern.ch/record/2667017.

[6] Miriam Calvo Gomez et al. *A tool for γ/π^0 separation at high energies*. Tech. rep. LHCb-PUB-2015-016. CERN-LHCb-PUB-2015-016. Geneva: CERN, Aug. 2015. URL: https://cds.cern.ch/record/2042173.

[7] Maureen Caudill. "Neural Networks Primer, Part I". In: *AI Expert* 2.12 (Dec. 1987), pp. 46–52. ISSN: 0888-3785. URL: http://dl.acm.org/citation.cfm?id=38292.38295.

[8] CERN. *About CERN*. 2015. URL: https://home.cern/about.

[9] S. Chatrchyan et al. "The CMS Experiment at the CERN LHC". In: *JINST* 3 (2008), S08004. DOI: 10.1088/1748-0221/3/08/S08004.

[10] Tianqi Chen. *Introduction to Boosted Trees*. 2014. URL: https://homes.cs.washington.edu/~tqchen/data/pdf/BoostedTree.pdf.

[11] Geant4 Collaboration. *Geant4: A simulation toolkit*. Dec. 2017. URL: http://geant4.web.cern.ch/.

[12] LHCb Collaboration. *LHCb Tracker Upgrade Technical Design Report*. Tech. rep. CERN-LHCC-2014-001. LHCB-TDR-015. Feb. 2014. URL: https://cds.cern.ch/record/1647400.

[13] Gloria Corti. *LHCb Simulation(s)*. Apr. 2014. URL: https://lhcb-comp.web.cern.ch/lhcb-comp/Simulation/Tutorial/MCInLHCb-GCorti-UKStudents-20140428.pdf.

[14] CERN COURIER. *LHCb improves trigger in Run 2*. Sept. 2015. URL: http://cerncourier.com/cws/article/cern/62495.

[15] Lyndon Evans and Philip Bryant. "LHC Machine". In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08001–S08001. DOI: 10.1088/1748-0221/3/08/s08001. URL: https://doi.org/10.1088%5C%2F1748-0221%5C%2F3%5C%2F08%5C%2Fs08001.

[16] Lyndon Evans and Philip Bryant. "LHC Machine". In: *JINST* 3 (2008), S08001. DOI: 10.1088/1748-0221/3/08/S08001.

[17] Python Software Foundation. *The Python Standard Library*. 2018. URL: https://docs.python.org/2/library/index.html.

[18] *Framework TDR for the LHCb Upgrade: Technical Design Report*. Tech. rep. CERN-LHCC-2012-007. LHCb-TDR-12. Apr. 2012. URL: https://cds.cern.ch/record/1443882.

[19] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[20] *Keras Library*. URL: https://keras.io.

[21] I. Kisel. "Event reconstruction in the CBM experiment". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 566.1 (2006). TIME 2005, pp. 85–88. ISSN: 0168-9002. DOI: https://doi.org/10.1016/j.nima.2006.05.040. URL: http://www.sciencedirect.com/science/article/pii/S0168900206008151.

[22] *LHCb calorimeters: Technical Design Report*. Technical Design Report LHCb. Geneva: CERN, 2000. URL: https://cds.cern.ch/record/494264.

[23] Ingo Lütkebohle. *Large Hadron Collider beauty experiment*. 2009. URL: https://lhcb-public.web.cern.ch/lhcb-public/Welcome.html.

[24] Michael Nielsen. *Neural Networks and Deep Learning*. 2015. URL: %5C%5Chttp://neuralnetworksanddeeplearning.com.

[25] K.A. Olive. "Review of Particle Physics". In: *Chinese Physics C* 38.9 (Aug. 2014), p. 090001. DOI: 10.1088/1674-1137/38/9/090001. URL: https://doi.org/10.1088%5C%2F1674-1137%5C%2F38%5C%2F9%5C%2F090001.

[26] Daniel Hugo Cámpora Perez. *Tracking Wheels*. Private Communication.

[27] Daniel Shiffman. *The Nature of Code: Simulating Natural Systems with Processing*. Ed. by Llc Theoklesia. Dec. 2012.

[28] Xabier Cid Vidal and Ramon Cid Manzano. *Taking a closer look at LHC*. 2015. URL: https://www.lhc-closer.es/taking_a_closer_look_at_lhc/1.home.