CENTRAL UNIVERSITY OF KARNATAKA

SCHOOL OF PHYSICAL SCIENCES

DEPARTMENT OF PHYSICS

Machine learning based event classification for INO ICAL prototype stack

Project report submitted by

Sourav Dutta
Registration Number: 2019MPH25

in partial fulfillment for the requirements of the degree of
Master of Science in Physics

Date of Submission: 06/06/2021

Supervised by: Dr. Deepak Samuel

Declaration

This project work was done by Sourav Dutta bearing registration no. 2019MPH25 from Department of Physics, Central University of Karnataka under my guidance at Central University of Karnataka, Gulbarga from 25/1/2021 to 6/6/2021.

Name of Supervisor: Dr. Deepak Samuel
Designation: Assistant Professor
Department of Physics
Central University of Karnataka

Signature with date

Acknowledgements

First of all, I would like to express my sincere gratitude to my supervisor Dr. Deepak Samuel who guided me in all stages of this project from choosing this topic to obtaining the results. He is a great person who has brought the best out of me by boosting my spirit. He has always motivated me in maintaining high morale and confidence whenever I was stuck with a problem. I am also grateful to Dr. Bharat Kumar, Dr. Rajeev Shesha Joshi and Dr. Suchismita Sahoo for their untiring support and motivation throughout my project. My experience of programming languages and interest in detector physics was the primary motivation to take up this project topic. I would like to thank all the members and my seniors of the HEP lab of CUK for their continuous support and guidance and also for sharing their knowledge with me. Last but not the least, I thank my parents and my brother for their affection and moral support.

Abstract

The India based Neutrino Observatory is a project, which aims to build an underground laboratory to study atmospheric neutrinos using a 50 kiloton Iron Calorimeter detector(ICAL) in a 1300 meter deep cave at Theni, Tamil Nadu, India. The primary goal of this observatory is to make a precise measurement of atmospheric neutrino oscillation parameters like mixing angle and the mass square difference and more importantly addressing the mass hierarchy problem. The Resistive Plate Chambers are being used as the active detector element in the INO-ICAL. As a part of its R & D program, a prototype detector stack consisting of 12 layers of glass Resistive Plate Chambers (RPCs) of size 1 m × 1 m has been set up at Tata Institute of Fundamental Research (TIFR), Mumbai. This detector stack is being used as a test-bench of the INO-ICAL. Cosmic muons are monitored continuously in this detector stack. Many cosmic muon studies like measurement of zenith angle distribution and cosmic muon intensity were performed using this stack. In this report, the result of analysis with simulated datasets is presented for muon multiplicity estimation using this stack and the result of analysis with real datasets is presented for event classification using this stack. Here we propose to use a machine learning-based algorithm to classify events as single muon, multi muon and noisy events. For this purpose, simulated datasets have been generated which have been used to train the machine and to test the efficiency as well. We have shown that the trained model can be used satisfactorily to classify the multi muon events, which were earlier regarded as noise.

Keywords: INO-ICAL, machine learning, event classification

Table of contents

List of figures **6**

List of tables **8**

1 Introduction **1**

1.1 Historical Background 1

1.2 India-based Neutrino Observatory 1

1.3 Iron CALorimeter detector (ICAL) 2

1.4 Prototype detector stacks 3

2 Prototype detector stack at TIFR, Mumbai **5**

2.1 Working principle of RPC 6

3 Present data analysis routine **8**

3.1 Straight Line Fit (SLF) algorithm 8

3.2 Failure of the SLF algorithm 9

4 Introduction to Machine learning **12**

4.1 Different types of learning algorithms 13

4.2 Training and Testing a Supervised Learning Algorithm 13

4.3 The Loss Function, Gradient Descent, and Regularisation 14

4.4 Gradient Boosted Decision Trees 15

4.4.1 Extreme Gradient Boosting (XGBoost) 16

5 Data Generation 17

5.1 Data generation algorithm 18

6 Data Analysis 22

6.1 Machine learning-based algorithm 22

6.1.1 Efficiency of Machine learning algorithm 23

7 Results 28

7.1 Analysis of real dataset 29

7.1.1 Predicted single muon events 29

7.1.2 Predicted multi-muon events 29

7.1.3 Predicted noisy events 29

7.1.4 Single muon events wrongly predicted as noisy events 33

7.1.5 Multi-muon events wrongly predicted as noisy events 33

7.2 Current status 33

7.3 Computational time 33

8 Conclusions and Future Scope 38

References 39

List of figures

1.1 Proposed site of the India-based Neutrino Observatory (INO) with 50 kilo ton ICAL (inset). The picture is taken from [7] 2

1.2 Schematic view of 50 kt ICAL. The picture is taken from [10]. The copper coils are shown in brown. RPCs (blue) are inserted into the air gaps between the iron plates (gray). 3

2.2 Schematic diagram of an RPC (transverse view). 6

3.1 Flowchart showing the sequence of algorithm used for SLF. The letters indicate the condition that was used to branch. Picture taken from [18] 10

3.2 A sample simulated cosmic muon track along with other additional hits. The actual track is 3-dimensional and this view shows only one of the projections. This event will be rejected in the SLF algorithm during the preconditioning stage itself. 11

5.1 Simulated events (clean tracks): (i) single muon event, (ii) event with multiplicity= 3, (iii) event with multiplicity= 5 19

5.2 A typical simulated multi-muon event before and after adding η, M$_s, M_n$. 21

6.1 Event classification accuracy for trigger condition: 4/12 24

6.2 Event classification accuracy for trigger condition: 5/12 25

6.3 Event classification accuracy for trigger condition: 6/12 26

6.4 Event classification accuracy for trigger condition: 7/12 27

7.1 Comparison of machine learning accuracy of event classification of a testing dataset with different training datasets 28

7.2 Some correctly predicted single muon events 30

7.3 Some correctly predicted multi-muon events 31

7.4 Some correctly predicted noisy events 32

7.5 Single muon events wrongly predicted as noisy events 34

7.6 Multi-muon events wrongly predicted as noisy events 35

7.7 Real event analysis 36

List of tables

3.1 Accepted and rejected layers according to the preconditioning for the SLF for the event shown in figure 3.2 11

5.1 Datasets 20

7.1 Computation time 36

Chapter 1

Introduction

1.1 Historical Background

The existence of neutrino was first proposed by Wolfgang Pauli in 1930 [15] to explain the continuous energy spectrum of beta decay. After that, through the inverse beta decay process, Frederick Reines and Clyde L. Cowan experimentally proved its existence [17]. From the experimental evidence of neutrino oscillation [11] and [8], we came to know that it has a nonzero mass. The non-zero mass of neutrino suggests physics beyond Standard Model which has far-reaching consequences in particle physics, astronomy, and cosmology.

Due to its low interaction cross-section $\sigma = 10^{-43} cm^2$ in weak interaction, high precision experiments are required to study neutrinos. Many experiments were conducted worldwide to detect and study neutrinos. Atmospheric neutrino events were recorded in the Kolar Gold Fields experiment, Karnataka in 1965 [9]. India Based Neutrino Observatory is another proposed project in this context, which aims to study neutrino oscillation parameters using atmospheric neutrinos.

1.2 India-based Neutrino Observatory

The India Based Neutrino Observatory is a proposed mega-science project of India whose primary goal is to study neutrino oscillation parameters and to precisely measure neutrino oscillation-parameters like mixing angle (θ_{23}) and the sign of the mass squared difference (Δm^2_{23}) and more importantly addressing the mass hierarchy problem. The project is a multi-institutional effort aimed at building an underground laboratory in a 1.3 km deep cavern of size 132 m × 26 m × 20 m at Pottipuram, Tamil Nadu, India (9°57′ N, 77°16′E). This rock cover will help in eliminating most of the cosmic ray background (mostly muons), allowing only neutrinos to reach the detector. The proposed site of INO has been shown in Fig. 1.1.

To study the neutrino oscillation parameters a 50 kiloton magnetized Iron-Calorimeter (ICAL) will be used. The ICAL will be able to measure the energy of the incoming particle, hence the name calorimeter.

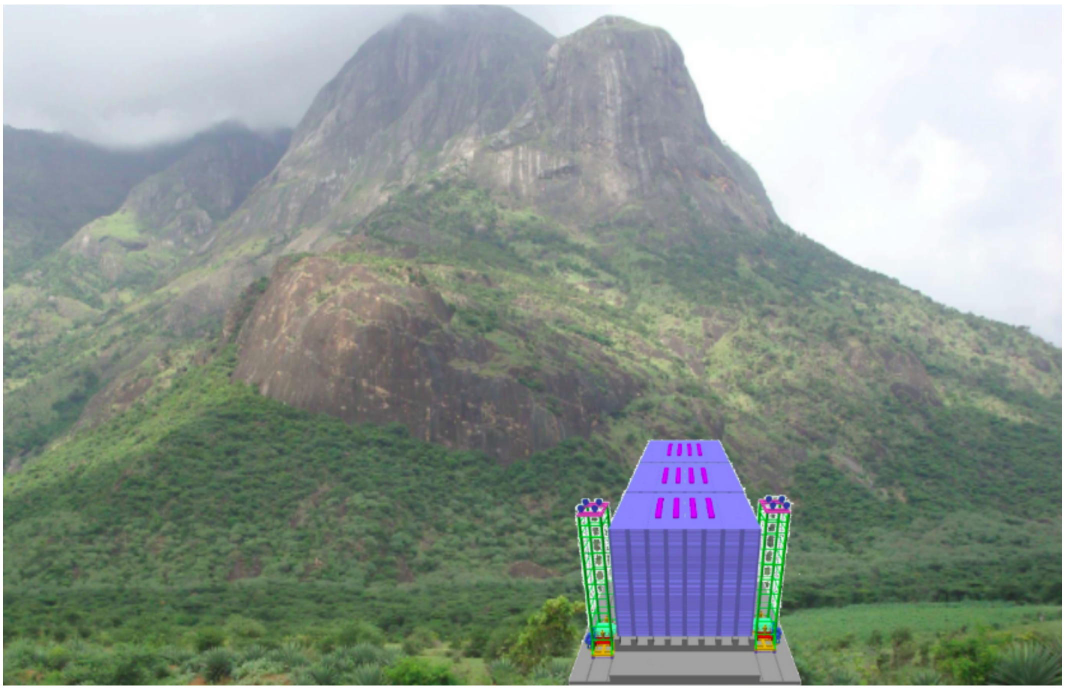

Fig. 1.1 Proposed site of the India-based Neutrino Observatory (INO) with 50 kilo ton ICAL (inset). The picture is taken from [7]

1.3 Iron CALorimeter detector (ICAL)

The ICAL is a sampling calorimeter detector made of Resistive Plate Chambers (RPCs) as the active detector elements. 30000 RPCs will be inserted into the air gaps between 150 layers of magnetized irons as shown in Fig. 1.2.

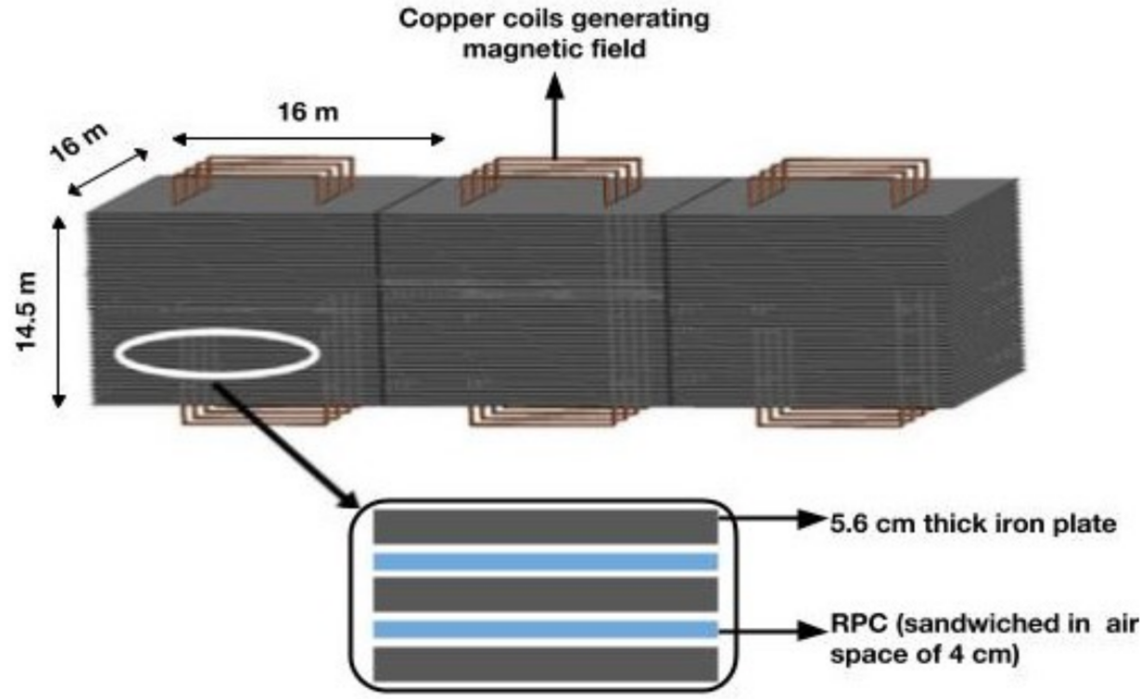
Copper coils generating magnetic field
16 m
16 m
14.5 m
5.6 cm thick iron plate
RPC (sandwiched in air space of 4 cm)

Fig. 1.2 Schematic view of 50 kt ICAL. The picture is taken from [10]. The copper coils are shown in brown. RPCs (blue) are inserted into the air gaps between the iron plates (gray).

The magnetic field of 1.3 T will be used here to separately identify muons and anti-muons from the opposite curvature of their tracks in ICAL. This will enable us to measure the atmospheric neutrino and anti-neutrino flux. ICAL will be primarily sensitive to atmospheric muon neutrinos in the 1–15 GeV energy range [10]. Muons will be generated in the ICAL by the charged current interactions of neutrinos with the iron nuclei. The produced muons will be detected by the RPCs, which will read out the strip hit position and timing information of muons from which the momentum and direction of neutrino will be reconstructed [10]. The RPCs offer a timing resolution of ∼1 ns which makes it effective in identifying up-going neutrinos from the down-going neutrinos [19]. This is important for the matter effects study in neutrino oscillation.

1.4 Prototype detector stacks

As a part of its research and development program, the INO collaboration has built many prototype stacks all over India in institutes like Tata Institute of Fundamental

Research Mumbai, Bhabha Atomic Research Centre (BARC) Mumbai, Saha Institute of Nuclear Physics (SINP), Kolkata, Variable Energy Cyclotron Centre (VECC), Kolkata, etc. A prototype detector stack consisting of 12 layers of RPCs of size 1 m × 1 m has been built at TIFR, Mumbai [Fig. 2.1]. It is continuously monitoring the cosmic muons [12]. A more detailed description of this detector stack is given in chapter 2.

Many cosmic muon studies like measurement of zenith angular distribution [14] and cosmic muon intensity [13] were performed using this stack. In this report, the simulation results of muon event classification using this stack are presented. At present, the cosmic muon analysis is being done by a straight-line fitting algorithm [12], because most of them are single muon events. This straight line fit considers interesting events, such as multi-muon events as noisy events and rejects them. But other experiments, such as the ALICE experiment of LHC [20] and the Frejus detector, which is located at the centre of the Frejus highway tunnel connecting Modane (France) to Bardonecchia (Italy) under the Alps [3], has shown the presence of muon bundles in cosmic ray composition in the high energy (PeV) range, indicating the presence of hevy nuclei in cosmic radiation. Therefore, in this study, we propose to classify and identify interesting events, such as multi-muon events using a machine learning-based algorithm. The machine is trained using simulated datasets and this trained model is used to predict real muon events. In this report, the results of the analysis with real data set are presented.

In chapter 2, a brief overview of the prototype stack at TIFR Mumbai is given, explaining its working principle.
Chapter 3 briefly explains the present data analysis routine and its failure in predicting multi muon events.
Chapter 4 briefly explains the underlying concepts in machine-learning algorithms.
Chapter 5 describes the data generation techniques and different data sets used for the analysis.
Chapter 6 provides the performance of the deep learning-based algorithm on the generated datasets and their predicted results on real datasets.
Chapter 7 concludes with the findings of this study.

Chapter 2

Prototype detector stack at TIFR, Mumbai

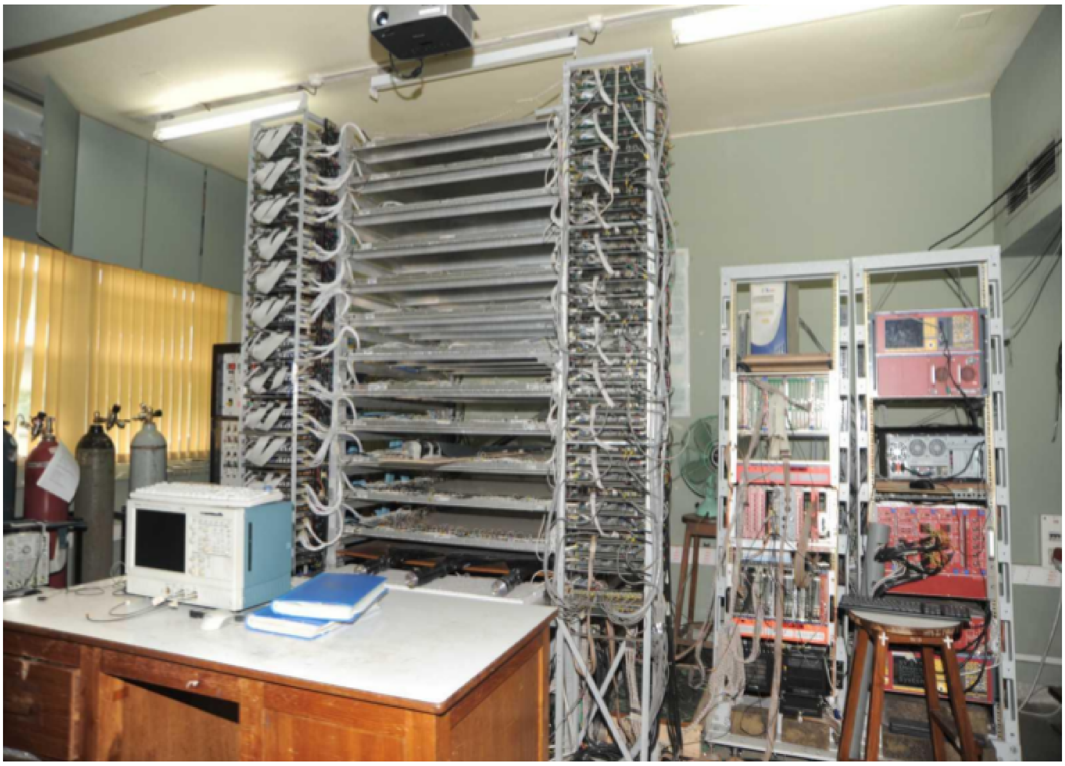

Fig. 2.1 The prototype detector stack at TIFR, Mumbai. The picture is taken from [7]. It has 12 layers, each layer consists of a Resistive plate chamber detector packed in an aluminum case. On either side of the detector stack the related electronics are shown.

The INO ICAL detector will be using about 30000 resistive plate chambers (RPC) detectors. So, in order to study them, a lot of R & D is pursued in different institutions in India. A prototype detector stack consisting of twelve layers of glass Resistive Plate Chambers (RPCs) of size 1 m × 1 m, has been set up at TIFR, Mumbai [2].

This detector stack is being used as a test-bench to study the long term performance and stability of the INO-ICAL and also for the development of related hardware and software, since 2008.

2.1 Working principle of RPC

RPCs form the active detector component of the prototype detector shown in Fig. 2.1. RPC is a gas-based ionization detector, comprising two glass plates of dimensions 1 m × 1 m × 2 mm which are held together by poly-carbonate buttons to maintain a 3 mm gap between them (shown in Fig. 2.2). Inside the gap, a non-flammable gas mixture (R134a ($1,1,1,2-$Tetrafluoroethane)-95.2 %, Isobutane ($2-$methylpropane)-4.5%, SF_6-0.3% in avalanche mode of operation) is flown. R134a is used here as the ionizing gas. Isobutane is used as the quenching gas for photon and SF6 being electro-negative, used as the quenching gas for electron [19].

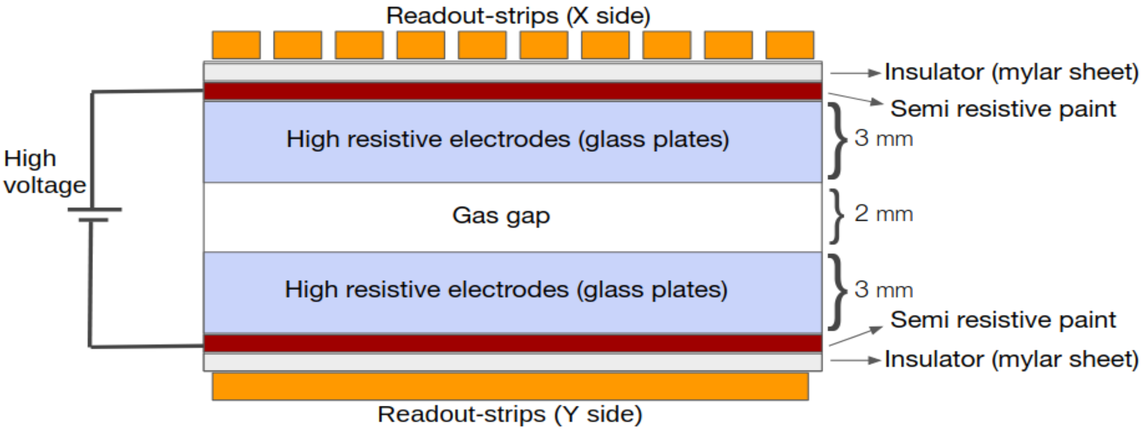

Fig. 2.2 Schematic diagram of an RPC (transverse view).

A high voltage of about 10 kV is applied uniformly across the glass plates through a resistive coating (graphite) on their outer surfaces as shown in Fig. 2.2. A muon crossing the detector ionizes a few of the gas molecules inside the RPC. The high electric field on either side of RPC accelerates the primary electron-ion pairs leading to further ionizations, resulting in the formation of an avalanche. The avalanche formation will result in a local discharge of electron. This discharge will be only in a small area due to the high resistivity of the glass plates and the quenching property of the gas. The propagation of charges due to avalanche induces a signal on either side of RPC.

To detect the induced signal, RPC is sandwiched between two honeycomb copper pickup panels orthogonal to each other. Each side consists of 32 parallel copper strips of width 28 mm separated by 2 mm gap between them, from which the X-Y co-ordinate of a muon is accessed. The Z-axis is accessed form the RPC layer number. In this way, the information of a muon passing through the detector is accessed from its strip hit position and timing information in the detector. The cosmic muon data collected from the detector is recorded after passing the valid trigger condition in the data acquisition system (DAQ). The TIFR prototype detector stack continuously receives cosmic particles each second. It has a trigger rate of 10 Hz, which implies, almost 36000 muons can be recorded per hour. These stacks are used for testing detector response and electronic subsystems. For this purpose, SLF algorithm is used since these stacks do not have magnetic field. A details discussion on the signal processing and DAQ can be found in [4]. A layer of 2 mylars of thickness 100 μm is placed between the graphite layer and the pickup panel to provide insulation.

The main advantage of using RPC is that it has very high efficiency: more than 90% and high timing resolution of about 1 ns.

Chapter 3

Present data analysis routine

At present, the straight line fitting (SLF) algorithm is used for track reconstruction in the TIFR prototype stacks.

3.1 Straight Line Fit (SLF) algorithm

When a muon passes through a RPC, only the corresponding strips (X and Y sides) that they pass through are expeceted to produce a signal [18]. A SLF is straightforward to implement in such cases. The SLF provides the slope and intercept of the two projections (i.e, the X and Y side strip hits) from which the 3-dimensional trajectory of the particle can be reconstructed. However, the presence of outliers in the form of additional hits in the detector, apart from the ones created by the particle, affects the fit parameters. The additional hits may arise due to detector or electronic noise. Hence, preconditioning the data before SLF is often necessary to remove the outliers or to make SLF less sensitive to outliers. The following cuts are applied before the SLF:

- **Condition a:** If a layer has more than 2 hits, it is not considered for the SLF.
- **Condition b:** If a layer has exactly 2 hits and the hits are separated by a distance equivalent to 2 strip-widths, the layer will not be considered for the SLF. Else, the average the two hits is taken as the hit position for that layer.
- **Condition c:** Once the above layer-level rejection is done, if the number of accepted layers is less than 5, the event will be rejected and the track will be classified as "not reconstructed".

- **Condition d:** If the event is accepted, a SLF is done to the hits obtained using the criteria mentioned above and the fit parameters are used to reconstruct the track.
- **Condition e:** It may happen that despite the above cuts, there might still be outliers which bias the fit parameters (also reflected in the χ^2 value). In order to reduce such bias, all hit points that are away from the fitted track by more than 2 strip-widths are removed (while retaining the layer in which the hit is less than 2 strip-widths from the first SLF) and a second SLF is made if the number of remaining layers is more than 2.
- **Condition f:** If the number of remaining layers is less than 2, the event is rejected and the track is classified as "not reconstructed".

The sequence of the algorithm has been explained with a flowchart in figure 3.1.

3.2 Failure of the SLF algorithm

The above cuts have been tested and the performance of the SLF after this filtering has been satisfactory for most of single muon event analysis. However, the SLF fails in cases where more noise hits are present alongside the actual hits. Simiarly, the SLF also wrongly predicts a muon event as a noisy event. The figure 3.2 shows a sample cosmic muon track (simulated) along one of the projections. Table 3.1 shows a summary of the accepted and rejected layers after the preconditioning cuts described above. It is evident that since 8 layers are rejected, the event is rejected in accordance with condition c and the track is tagged as "not reconstructed". Nevertheless, a visual survey of the track shows that there are clearly two tracks amidst the noise hits. What makes the SLF fail where the visual identification of the track is possible without much effort was the motivaton behind this work.

As seen from these plots, such multi-muon events may be interesting because these imply the presence of muon bundles in the cosmic radiation, implying the presence of heavy nuclei in cosmic radiation. Hence such events are required. For this purpose, machine learning algorithm needs to be implemented since it is an innovative idea which does not need explicit programming.

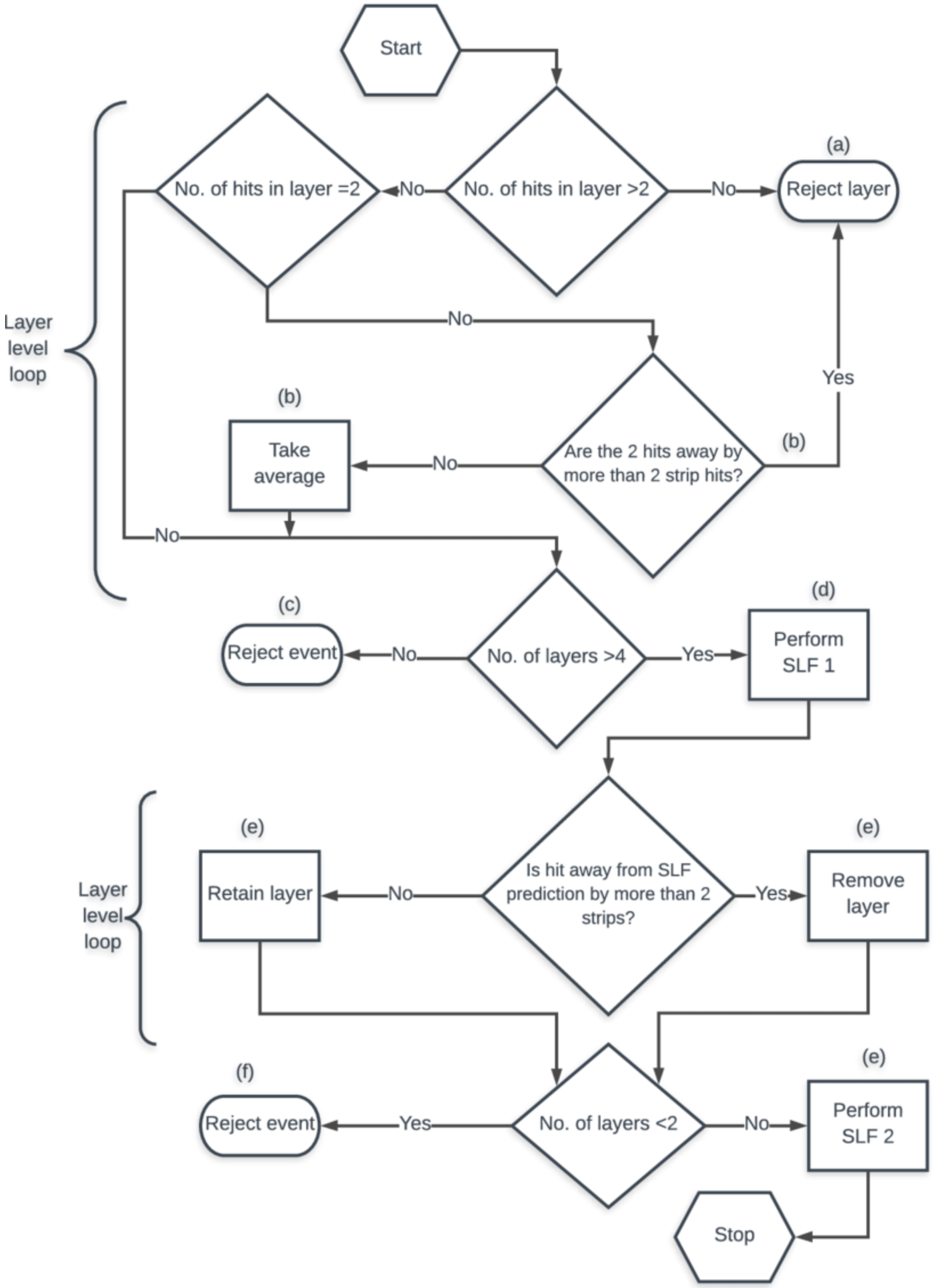

Fig. 3.1 Flowchart showing the sequence of algorithm used for SLF. The letters indicate the condition that was used to branch. Picture taken from [18]

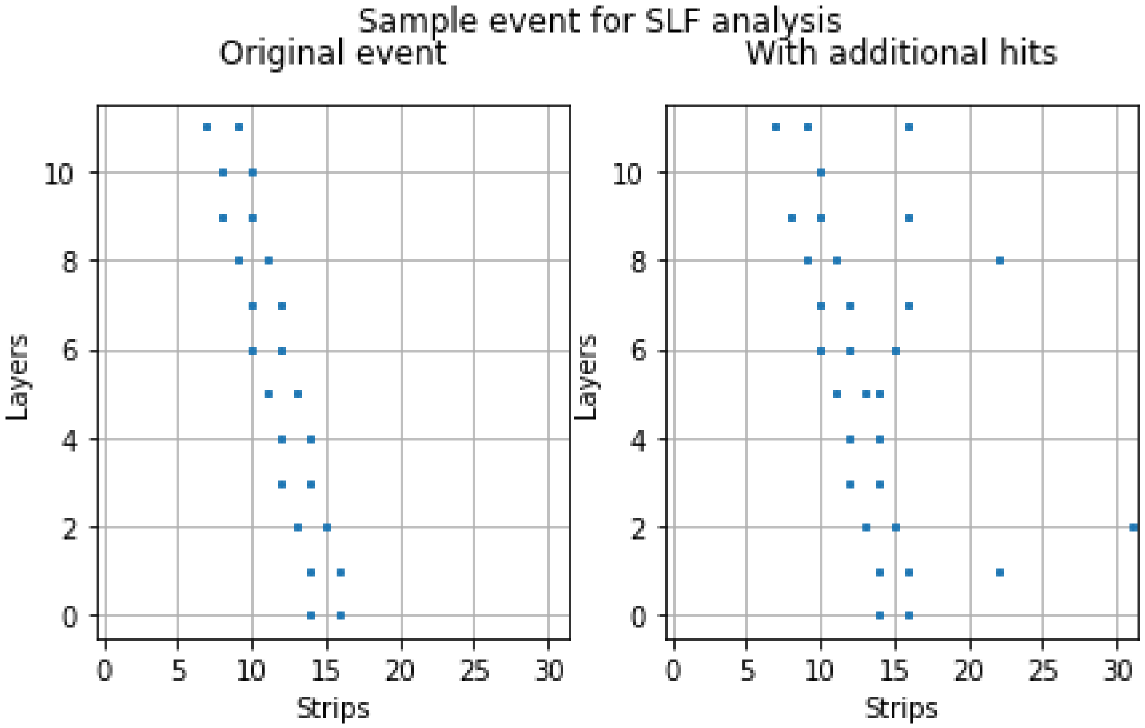

Fig. 3.2 A sample simulated cosmic muon track along with other additional hits. The actual track is 3-dimensional and this view shows only one of the projections. This event will be rejected in the SLF algorithm during the preconditioning stage itself.

Layer	Status	Condition
0	Accepted	2 hits away by less than 2
1	Rejected	a
2	Rejected	a
3	Accepted	2 hits away by less than 2
4	Accepted	2 hits away by less than 2
5	Rejected	a
6	Rejected	a
7	Rejected	a
8	Rejected	a
9	Rejected	a
10	Accepted	single hit
11	Rejected	a

Table 3.1 Accepted and rejected layers according to the preconditioning for the SLF for the event shown in figure 3.2

Chapter 4

Introduction to Machine learning

Machine learning (ML) algorithms allow a computer to make decisions on its own by learning the correlations and patterns from large data without being programmed explicitly. ML finds a pattern in the data, followed by adapting the changes and improving the performance with experience. The last decade has seen a lot of application of machine learning across both academia and industry, which has been accelerated by several reasons, including:

- the amount of data, which is being generated in everyday life has increased exponentially,
- most of the generated data has become easily accessible over the internet, and
- the analytical power of modern computers has increased immensely.

In the field of experimental particle physics, machine learning is playing an important role in the development of new theories of physics by observing rare events. Machine learning has been applied to several problems in particle physics research, beginning with applications to high level physics analysis in the 1990s and 2000s, followed by an explosion of applications in particle and event identification and reconstruction in the 2010s [1]. It is one of the promising techniques to solve multi-particle challenges [6] of high energy physics. It is also used in particle and event identification, track reconstruction, track fitting [18] etc.

4.1 Different types of learning algorithms

The machine learning algorithms can be discussed into three categories:

- **Supervised Learning**: The trained model is built from labeled data and predicts the labels of unlabelled data. Such algorithms can be further classified as regression and classification algorithms. The output of a regression algorithm is usually a real-valued number (e.g. predicting the energy or momentum of a particle) while classification algorithms classify the data into one of the given classes (e.g. to predict the particle multiplicity of an event).
- **Unsupervised Learning**: This algorithm works with unlabelled data and learns its characteristics from the data. Clustering and association are two common examples of unsupervised problems.
- **Semi-supervised algorithm**: Algorithms learn from the data which is partly labeled and make predictions.

4.2 Training and Testing a Supervised Learning Algorithm

The supervised learning algorithm needs to be trained and tested. This process can normally be discussed in the following way:

1. **Training**: The machine learning algorithm finds a pattern from a given data set. This data set is called the “training data”. Training data contains both features and labels. Given the training data to the algorithm, the algorithm will generate a function (f) that best fits the data. Training the model is the most human and CPU-time consuming step [1].
2. **Testing**: After training, the algorithm is provided with a new data set called “testing data”, which contains only the “features”. The label of this data set is evaluated with the help of the model, which is generated during training. Testing a trained model is relatively inexpensive [1].

4.3 The Loss Function, Gradient Descent, and Regularisation

During training, machine learning algorithms generate a function f that best fits the data given. The function can be formulated as:

$$f(x,\theta) = \hat{y} \tag{4.1}$$

where the known properties of the given data which make up the vector x are known as the "input features". The function f, which is constructed by the machine learning algorithm is referred to as the "predictor" [6]. The predictor (f) evaluates a prediction, $\hat{y}$, of the target property or label (y), based on the input features (x). Constructing the function normally means determining the optimal set of parameters θ.

The performance of the algorithm in both training and testing stages is determined by a "loss function" [6], which calculates the difference between the label value y and its prediction $\hat{y}$ for the entire data set. An example of the loss function is the mean squared error:

$$E(Y,\hat{y}) = \frac{1}{2N}\sum_{i=1}^{N}(\hat{y}^{(i)} - y^{(i)})^2 \tag{4.2}$$

Using equation 4.1, the above equation can be rewritten as:

$$E(Y,X,\theta) = \frac{1}{2N}\sum_{i=1}^{N}[f(x^{(i)},\theta) - y^{(i)}]^2 \tag{4.3}$$

where the function E is the loss function, which depends on Y, X, θ, where Y is the target value (labels) for the entire data set, ŷ is the prediction of the label across all the elements in the data set, and X is the matrix representing all the features for the data set. The right-hand side of equation 4.3 is the average of the mean squared error from all N elements in the entire data set, where the superscript denotes the i^{th} data element.

The parameter θ is iteratively updated while training by a process known as "gradient descent"[6]. It uses the differential of the loss function with respect to each element in θ, to update θ in a way which minimizes the loss function, E. This can be mathematically expressed as follows:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} E(Y, X, \theta) \tag{4.4}$$

Using equation 4.3, equation 4.4 can be rewritten as:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} \frac{1}{2N} \sum_{i=1}^{N} [f(x^{(i)}, \theta) - y^{(i)}]^2 \tag{4.5}$$

$$Or, \theta_j := \theta_j - \frac{\alpha}{N} \sum_{i=1}^{N} [f(x^{(i)}, \theta) - y^{(i)}] \frac{\partial}{\partial \theta_j} f(x^{(i)}, \theta) \tag{4.6}$$

In the above equations, α is referred to as the "learning rate", which controls the speed of convergence. α helps the loss function to reach the global minima, by carefully avoiding the local minimum. In machine learning, this can often happen when an algorithm finds a minimum that is unique to a particular training data set. This problem is called "over-fitting". To avoid over-fitting, the "regularisation" technique is often used, where instead of using the loss function E, a new loss function $J(Y, X, \theta)$ can be used:

$$J(Y, X, \theta) = E(Y, X, \theta) + \frac{\lambda}{2} \sum_{j=1}^{N} \theta_j^2 \tag{4.7}$$

In equation 4.7, the second term is the regularisation term. This term does not allow the model to rely too heavily on any one particular θ_j. The parameter λ controls the strength of this process. The update rule can be defined using the function J in place of E in equation 4.4:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(Y, X, \theta) \tag{4.8}$$

4.4 Gradient Boosted Decision Trees

So far the discussion about machine learning has been in general terms. This section will specifically discuss the algorithm used in this work: Extreme gradient boosted decision trees. Gradient boosted decision trees (GBDTs) is one of the "ensemble" machine learning techniques. It is an ensemble of shallow trees placed sequentially where each tree learns and improves the next one. In this way, it combines many weak learning methods to generate a much stronger one using a gradient boosting technique to minimize the loss function of the model, hence the name. GBDT involves 3 elements:

- A loss function to be optimized
- A weak learner to make predictions
- An additive model which will add weak learners to generate a much stronger one in such a way as will minimize the loss function.

Traditionally, GBDTs are slow because of its sequential nature in which each tree must be constructed and added to the model.

4.4.1 Extreme Gradient Boosting (XGBoost)

XGBoost is a scalable machine learning system for tree boosting. The system is available as an open source package. The impact of the system has been widely recognized in a number of machine learning and data mining challenges. XGBoost is an implementation of gradient boosting that is designed to improve its speed and performance. Extreme gradient boosting is one of the most powerful techniques to build predictive models. The algorithm was developed by Tianqi Chen and Carlos Guestrin [5]. XGBoost is a perfect combination of hardware and software optimization techniques to produce superior results using less computing resources in less amount of time. In addition to supporting all the key features of GBDT, it immensely decreases the computational time in the following ways:

- Parallelization in shallow tree construction using all the available CPU cores or GPU during training.
- Distributed Computing for training large models using a cluster of machines[6].

The most important factor behind the success of XGBoost is its scalability in all scenarios. The system runs more than ten times faster than existing popular solutions on a single machine and scales to billions of examples in distributed or memory-limited settings. The scalability of XGBoost is due to several important systems and algorithmic optimizations. These innovations include: a novel tree learning algorithm is for handling sparse data; a theoretically justified weighted quantile sketch procedure enables handling instance weights in approximate tree learning. Parallel and distributed computing makes learning faster which enables quicker model exploration [1].

Chapter 5

Data Generation

A machine learning (ML) algorithm learns from the training data. Therefore, the data used for training an algorithm should have all the features correlated to the problem and should be as realistic as possible. The number of samples in the training data should also be large enough for the ML model to predict results from any unseen dataset.

When a cosmic muon passes through the prototype detector (Fig. 2.1), it leaves behind its information in terms of strip hit pattern. A 3-dimensional track of a cosmic muon is displayed and recorded from its 2-dimensional projection in both xz and yz plane of the detector in terms of strip hit pattern. As the detector has 12 layers of RPCs and in each layer, there are 32 strips in both x and y side, a cosmic muon event can be simulated as two sparse matrices of dimension [12 × 32], each element representing a strip in a layer. The value one (1) is assigned to a strip which is having a hit and zero (0) elsewhere. In the real experiment, cosmic muon tracks are affected by many parameters of the detector like its efficiency, strip multiplicity, noise multiplicity, etc. To make the cosmic muon simulation more realistic, 3 factors are included [18] with the clean tracks. They are:

1. **Detector efficiency (η)**: This is the ratio of the number of cosmic muons that have gone through the detector to the number of cosmic muons that have given a signal in the RPCs. As previous studies have shown that RPCs have an efficiency (η) of around 90–95% [12], here we have considered $\eta = 95\%$.

2. **Strip hit multiplicity (M_s)**: The average number of strips fired due to an ionizing particle is called the cluster size or strip multiplicity (M_s), which depends

on the gain of the gas inside the RPCs. High cluster size affects the accuracy of the track localization. From previous studies, the average M_s of the RPCs for cosmic muon data is found to be ~ 1.5 [19], and therefore we have restricted M_s to have a minimum value of 1 and a maximum of 2.

3. **Noise hit multiplicity (M_n)**: Due to improper gas distribution inside RPCs or due to electronic noise produced by different sources, there might be uncorrelated noise hits in a layer. The maximum number of noise hits in a layer is called the noise hit multiplicity (M_n). In this study, M_n is restricted between 0 and 5 [18].

5.1 Data generation algorithm

1. In the TIFR prototype stack [Fig 2.1], as there is no magnetic field, cosmic muons follow straight-line trajectories. Therefore, cosmic muon data is simulated in a [12 × 32] sparse matrix in terms of its strip hit pattern, using the equation of a straight line:
$$s_x = m_x l_z + c_x \tag{5.1}$$
where s_x is the strip number in the x-side of the z^{th} layer (l_z) of the stack. A similar equation can be written to represent the y-side hits.

2. The strip number of a corresponding layer can be estimated from equation 1, if the slope (m_x) and intercept (c_x) are known. Due to the uniform angular distribution of cosmic muons flux, they are expected to come from any direction and intercept at any point on the detector. Thus tracks are generated using equation 1 with randomly generated slopes and intercepts.

3. Since the number of detector layers in the stack is 12 and each layer has 32 strips, the slope is constrained between -2.66 and +2.66 and the intercept is constrained between 0 and 31. Where the slope (m_x) is in units of strips per layer and the intercept (c_x) is the strip hit in the 0^{th} layer.

4. The cosmic muon can enter the detector at any layer and can come out of the detector at any layer. These layer numbers are also generated randomly.

5. Tracks are generated for both xz and yz planes and tracks to be used in the analysis were required to have the same number of hits in both xz and yz planes.

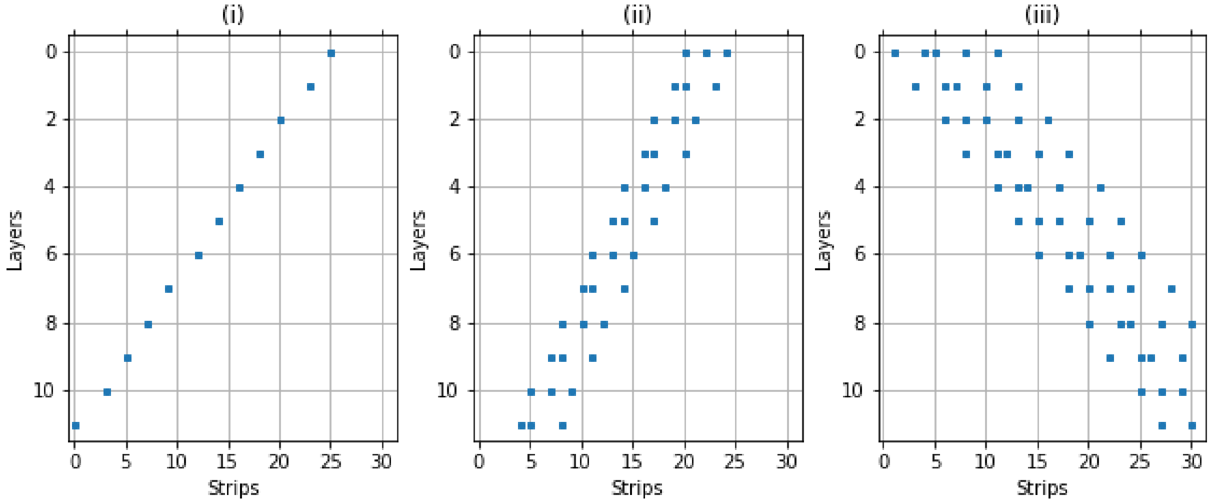

Fig. 5.1 Simulated events (clean tracks): (i) single muon event, (ii) event with multiplicity= 3, (iii) event with multiplicity= 5

6. The events in which a track passes through more than 3 trigger layers in both the views are considered for this study (In the real experiment, this is satisfied by setting a top-bottom coincidence trigger by a VME based trigger module).

7. Step 1 to step 6 together generate a single muon event. Multi-muon events are generated by repeating this entire process (Step 1 to step 6) multiple times. For muon bundles, the slope is kept constant and the intercept (c_x or the strip hit in the 0^{th} layer) is randomly generated.

8. To generate n number of events, Step 1 to step 7 is repeated for n number of times. The data sets are generated in such a way, that the number of events is distributed equally among different multiplicities. The entire process (step 1 to step 8) will generate a data set containing clean tracks (without noise). In Fig. 5.1, 3 simulated events (without noise) can be found.

9. To consider the detector efficiency (η), first, a particular η value is fixed between 0 and 100. Then another random number R_1 is generated between 0 to 100 corresponding to each strip hit. If R_1 is greater than η, the corresponding strip hit is removed from the event.

10. To consider the strip multiplicity (M_s), a uniform random number S1 is generated between 0 and 150 for every strip hit. If generated number S1 is between 0 and 50, one extra hit is added on the left side of the main strip hit. If S1 is greater

than 100 and less than 150, a hit is added on the right side of the main strip and if the generated random number is greater than 50 and less than 100, the hits corresponding to these numbers are allowed to remain the same.

11. To consider the noise hit multiplicity (M_n), some extra hits are added to each layer by generating a random integer N for each layer between 0 and M_n. Also, N random integers between 0 and 32 are generated and the corresponding strip numbers are set to 1. A typical multi-muon event with multiplicity 4 is shown in Fig 5.2 before and after adding the detector efficiency ($\eta = 95\%$), strip hit multiplicity ($M_s = 1.5$), noise hit multiplicity M$_n = 5$).

Based on the above mentioned criteria, four different types of datasets have been generated, each with different trigger conditions. Dataset-1 has 60000 events with 5000 events of multiplicity (1 to 12). In this dataset, the events have trigger condition 4/12, which means it must pass through 4 layers. Dataset-2 to Dataset-4 are having events with different trigger conditions. Trigger condition ensures that the muon track has passed through a minimum number of layers. In the real detector, this trigger generation is regulated by a VME based trigger module in the DAQ.

Dataset	Trigger condition	Total number of events	Number of events/multiplicity	Multiplicity
Dataset-1	4/12	60000	5000	1 to 12
Dataset-2	5/12	70000	5000	1 to 14
Dataset-3	6/12	55000	5000	1 to 11
Dataset-4	7/12	50000	5000	1 to 10

Table 5.1 Table summarizing the datasets used in this study.

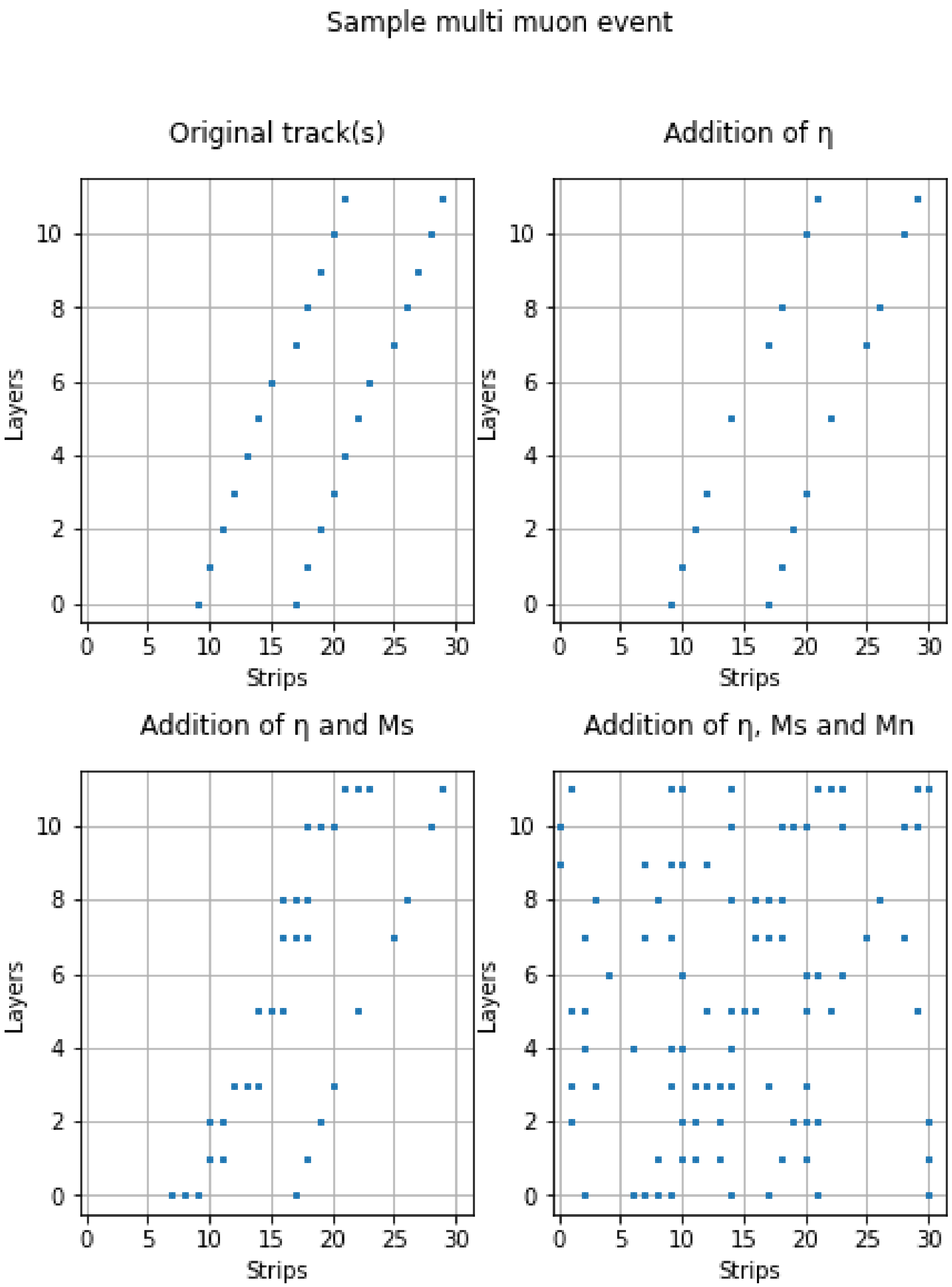

Fig. 5.2 A typical simulated multi-muon event before and after adding η, M_s, M_n.

Chapter 6

Data Analysis

In this study, the Extreme Gradient Boosting (XGBoost) algorithm has been used for data analysis. This chapter includes an overview of the algorithm, followed by a discussion about its advantages and shortcomings and the result of analysis of this algorithm.

6.1 Machine learning-based algorithm

The machine learning-based algorithm which is particularly used in this study is discussed in this section. The motivation behind using a machine learning-based algorithm for this study was the apparent failure of Hough transform based algorithm or straight line fit (SLF) algorithm at higher multiplicities, where the visual estimation of the multiplicity is possible up to a certain multiplicity without much effort. In this study to estimate the muon-multiplicity of an event, a python based classifier algorithm - XGBClassifier is used. This algorithm is available in the XGBoost module of the scikit-learn toolkit [16]. To create the boosted decision tree models, the XGBClassifier algorithm is trained with a data-set of table 5.1. For training, 80% of the data of a particular data-set is used. This training data contains both features and labels. In this study, the features are the strip hit pattern and the label is the multiplicity of an event. After training, given the test data (remaining 20% data of the same data-set) containing only the features, the model predicts the label (multiplicity) of an event and for the entire test data-set, the efficiency of the algorithm is calculated. Based on the performance of the algorithm, parameters like learning rate, max_depth and n_estimator are fine-tuned to get better efficiency.

- The **max_depth** is the maximum depth of an individual weak learner (shallow trees). **max_depth** is set between 2 and 8.
- The **learning_rate** controls the speed of convergence to find the global maxima, avoiding the local maxima. Typically **learning_rate** is set at about 0.1 or lower values. Smaller values of the **learning_rate** will require an addition of more trees. If the model is found to be over learning, the **learning_rate** can be decreased or the number of trees can be increased.
- The **n_estimator** is the number of weak learners (shallow tress) to be used. Depending on the size of the data, **n_estimator** (number of trees) is chosen between 100 and 10000.

After the fine-tuning of these parameters, a set of parameters is selected for a particular data-set, which gives better efficiency than others.

6.1.1 Efficiency of Machine learning algorithm

In this section the results of the analysis with the machine learning-based algorithm is summarized. The study has been done with 5 different data-sets having different trigger conditions. For this study, all the data-sets of table 5.1 have been used. Each data-set is generated for clean tracks, adding detector efficiency (η), adding η and strip hit multiplicity (M_s), adding η, M_s, and the noise hit multiplicity (M_n). The results of the analysis considering these effects corresponding to each trigger condition are discussed below graphically, where different colored tracks represent the following:

Blue curve:	Efficiency with clean tracks
Orange curve:	Efficiency after adding η on clean tracks
Green curve:	Efficiency after adding η, M_s on clean tracks
Red curve:	Efficiency after adding η, M_s and M_n on clean tracks

Dataset-1:

This dataset is simulated with a trigger condition 4/12 which means that tracks in an event have passed through a minimum of 4 layers in this data-set. The corresponding efficiency plot is shown in red in Fig. 6.1. For clean tracks, the single muon efficiency is almost 100%, it decreases to 99% after addition of detector efficiency η, to a little

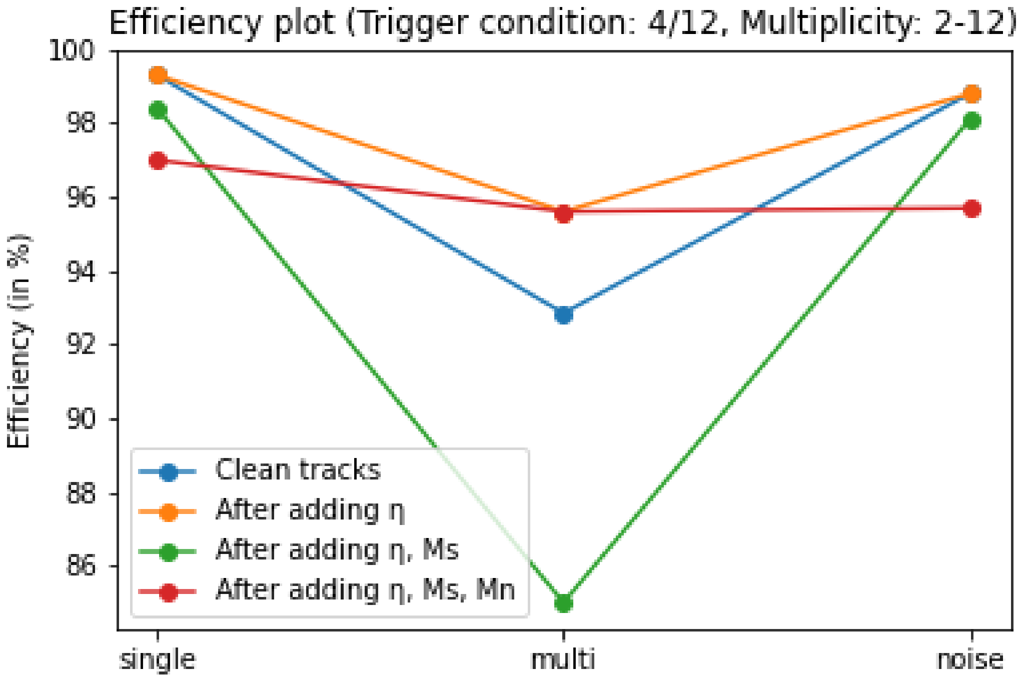

Fig. 6.1 Event classification accuracy for trigger condition: 4/12

more than 98% after addition of M_s, and almost 97% after addition of M_n. For multi muon events, the clean tracks have efficiency almost 93%, which increases to almost 96% after addition of η, and again decreases to almost 85% after addition of M_s, and increases to 96% after addition of M_n. For noisy events, the clean tracks have efficiency almost 98%, which increases to almost 99% after addition of η, and decreases to almost 98% after addition of M_s. The accuracy further decreases to 96% after addition of M_n, as is evident from the figure 6.1

Dataset-2:

This dataset is simulated with a trigger condition 5/12 which means that tracks in an event have passed through a minimum of 5 layers in this data-set. The corresponding efficiency plot is shown in red in Fig. 6.2. For clean tracks, the single muon efficiency is almost 100%, it remains the same after addition of detector efficiency η and M_s, and decreases to 99% after addition of M_n. For multi muon events, the clean tracks have efficiency almost 88%, which remains the same after addition of η, and then decreases to almost 82% after addition of M_s, and almost to 72% after addition of M_n. For noisy

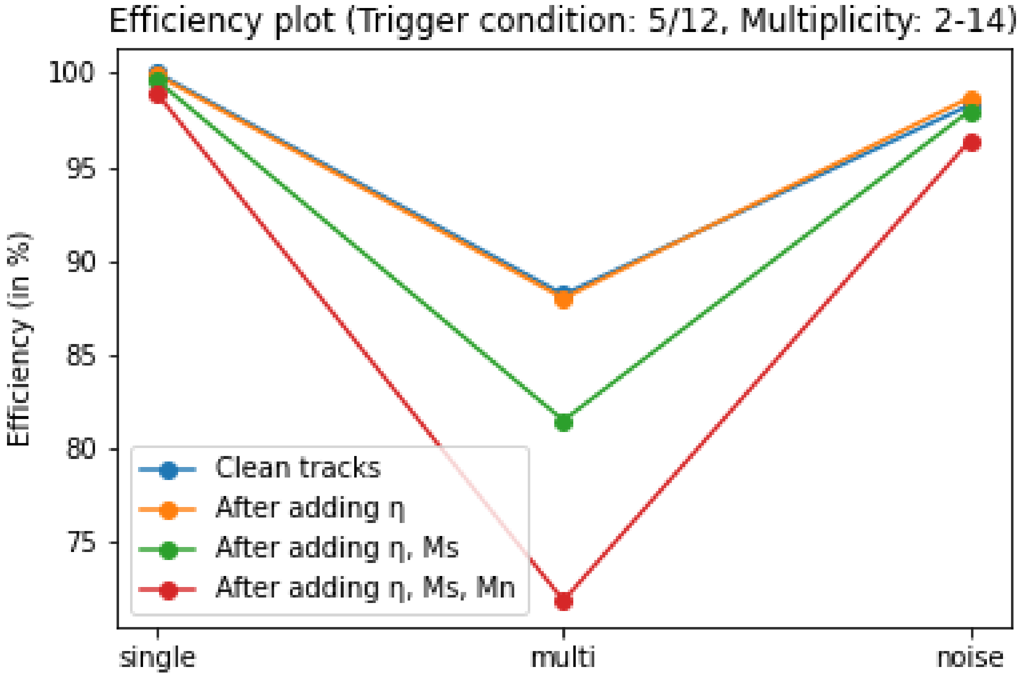

Fig. 6.2 Event classification accuracy for trigger condition: 5/12

events, the clean tracks have efficiency almost 98%, which increases to almost 99% after addition of η, and decreases to almost 98% after addition of M_s. The accuracy further decreases to 97% after addition of M_n, as is evident from the figure 6.2

Dataset-3:

This dataset is simulated with a trigger condition 6/12 which means that tracks in an event have passed through a minimum of 6 layers in this data-set. The corresponding efficiency plot is shown in red in Fig. 6.3. For clean tracks, the single muon efficiency is almost 100%, it remains same after addition of detector efficiency η, and then decreases to 99% after addition of M_s, and further to 98% after addition of M_n. For multi muon events, the clean tracks have efficiency almost 83%, which remains same after addition of η, it increases to almost 87% after addition of M_s, and further to 98% after addition of M_n. For noisy events, the clean tracks have efficiency almost 99%, which remains same after addition of η, and M_s. The accuracy further decreases to 96% after addition of M_n, as is evident from the figure 6.3

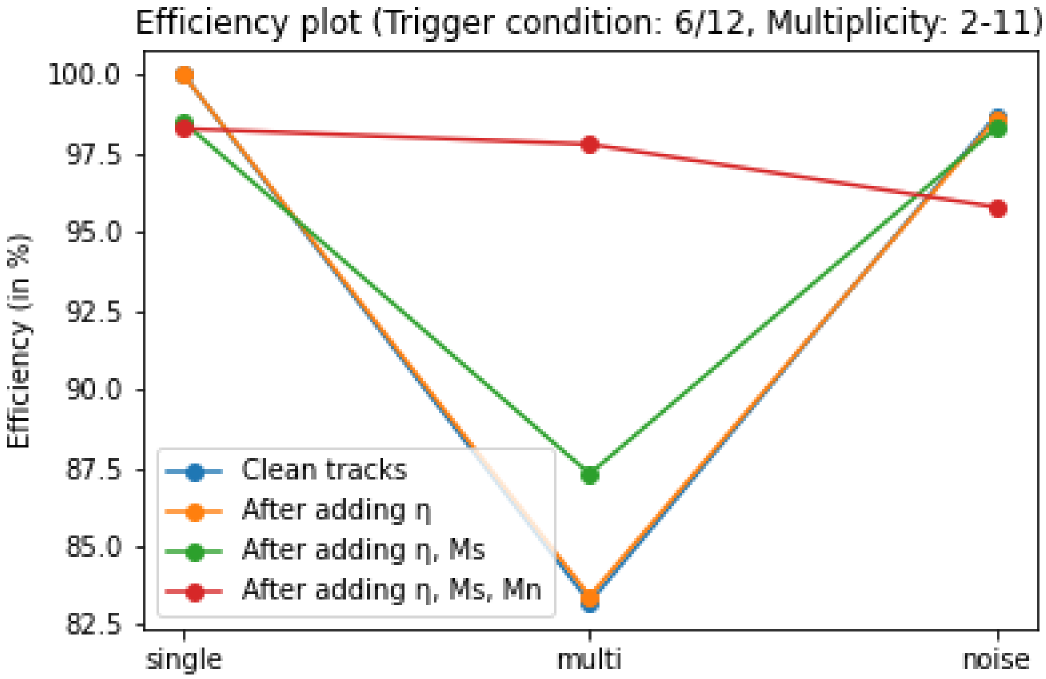

Fig. 6.3 Event classification accuracy for trigger condition: 6/12

Dataset-4:

This dataset is simulated with a trigger condition 7/12 which means that tracks in an event have passed through a minimum of 7 layers in this data-set. The corresponding efficiency plot is shown in red in Fig. 6.4. For clean tracks, the single muon efficiency is almost 100%, it remains same after addition of detector efficiency η, M_s, and M_n. For multi muon events, the clean tracks have efficiency almost 93%, which decreases to almost 78% after addition of η, and again increases to almost 79% after addition of M_s, and further to 98% after addition of M_n. For noisy events, the clean tracks have efficiency almost 99%, which remains same after addition of η, and M_s. The accuracy decreases to 97% after addition of M_n, as is evident from the figure 6.4

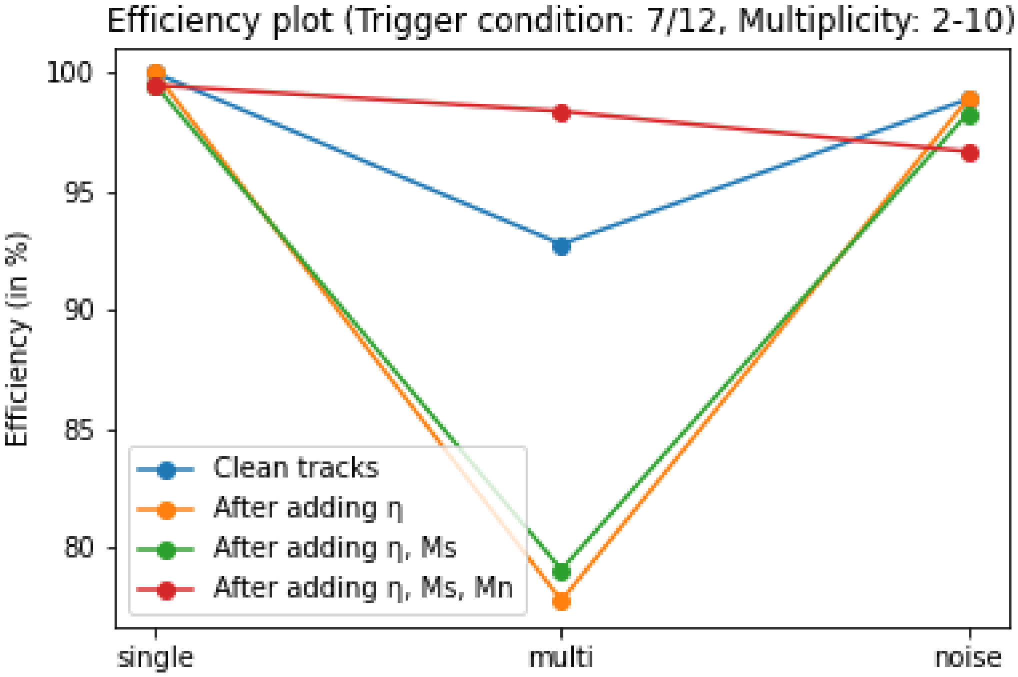

Fig. 6.4 Event classification accuracy for trigger condition: 7/12

Chapter 7

Results

In this chapter, the results of the analysis using machine learning algorithm is presented. The analysis using machine learning-based algorithm is done with data sets having different trigger conditions.

The blue, orange, green and red curves as shown in figure 7.1 are the efficiency of the machine learning algorithm using all the datasets of table 5.1.

It is evident from figure 7.1 that the efficiency of the machine learning-based algorithm is 99% for single muon events for the training dataset trigger 4/12, and increases to 100% for all the rest training datasets. For multi muon events, the accuracy is almost 98% for trigger 6/12 and 7/12. It decreases to 96% for trigger 4/12, and further to 88% for trigger 5/12. For noisy events, the accuracy is 99% for all the training datasets. From this comparative study, it is found that the machine learning algorithm is more accurate than the straight line fit (SLF) algorithm in predicting muon-multiplicity of

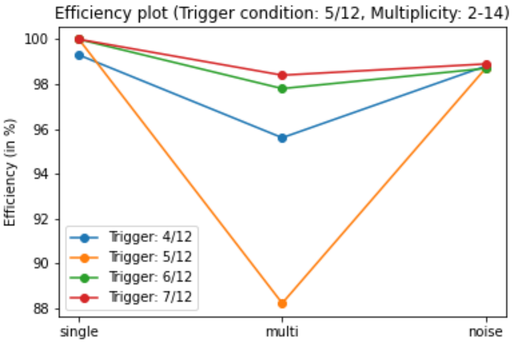

Fig. 7.1 Comparison of machine learning accuracy of event classification of a testing dataset with different training datasets

an event. From the study, the efficiency of the machine learning-based algorithm is found to be improving with increase in the number of layers in the trigger.

7.1 Analysis of real dataset

The trained model was used in predicting a sample dataset containing one lakh events. The dataset was recorded by the TIFR prototype detector [Figure 2.1]. The result is listed below:

- File name: INORUN_20151110_181937.txt
- Total events: 100000
- Single muon events: 25742 (25.75%)
- Multi muon events: 9820 (9.82%)
- Noisy events: 64438 (64.44%)

7.1.1 Predicted single muon events

Some of the events from the above dataset which are predicted as single muon events are shown in figure 7.2.

7.1.2 Predicted multi-muon events

Some of the events from the dataset which are predicted as multi muon events are shown in figure 7.3.

7.1.3 Predicted noisy events

Some of the events from the dataset which are predicted as noisy events are shown in figure 7.4.

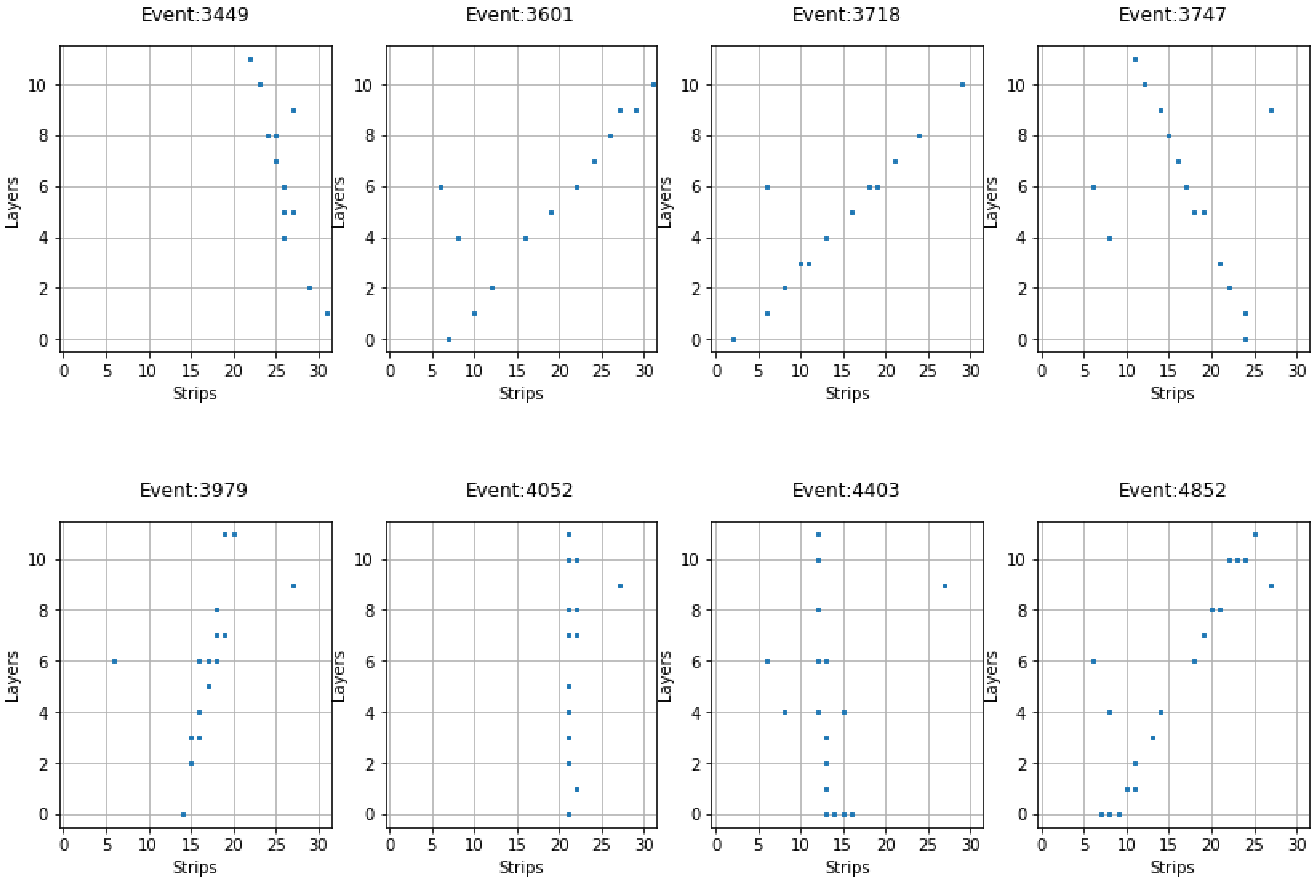

Fig. 7.2 Some correctly predicted single muon events

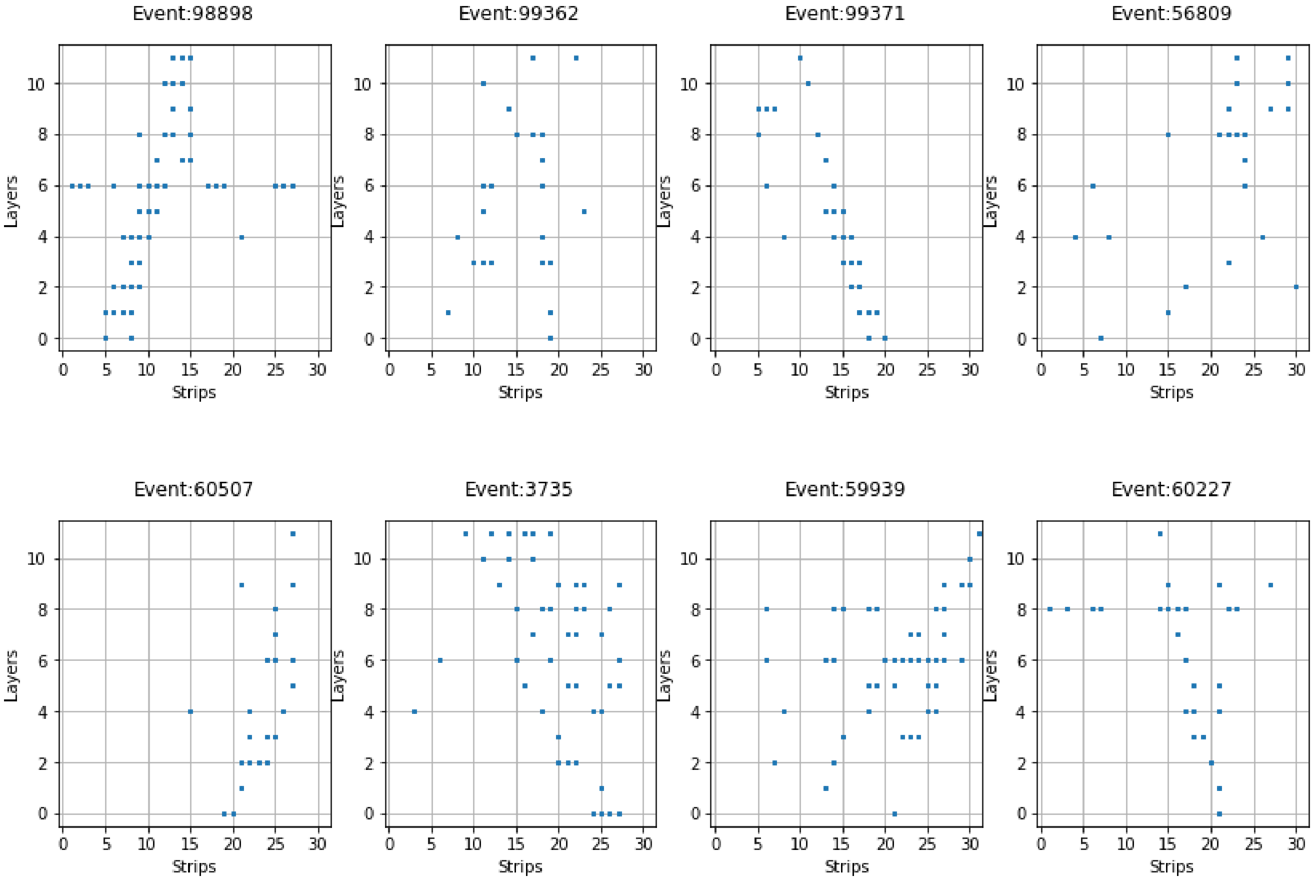

Fig. 7.3 Some correctly predicted multi-muon events

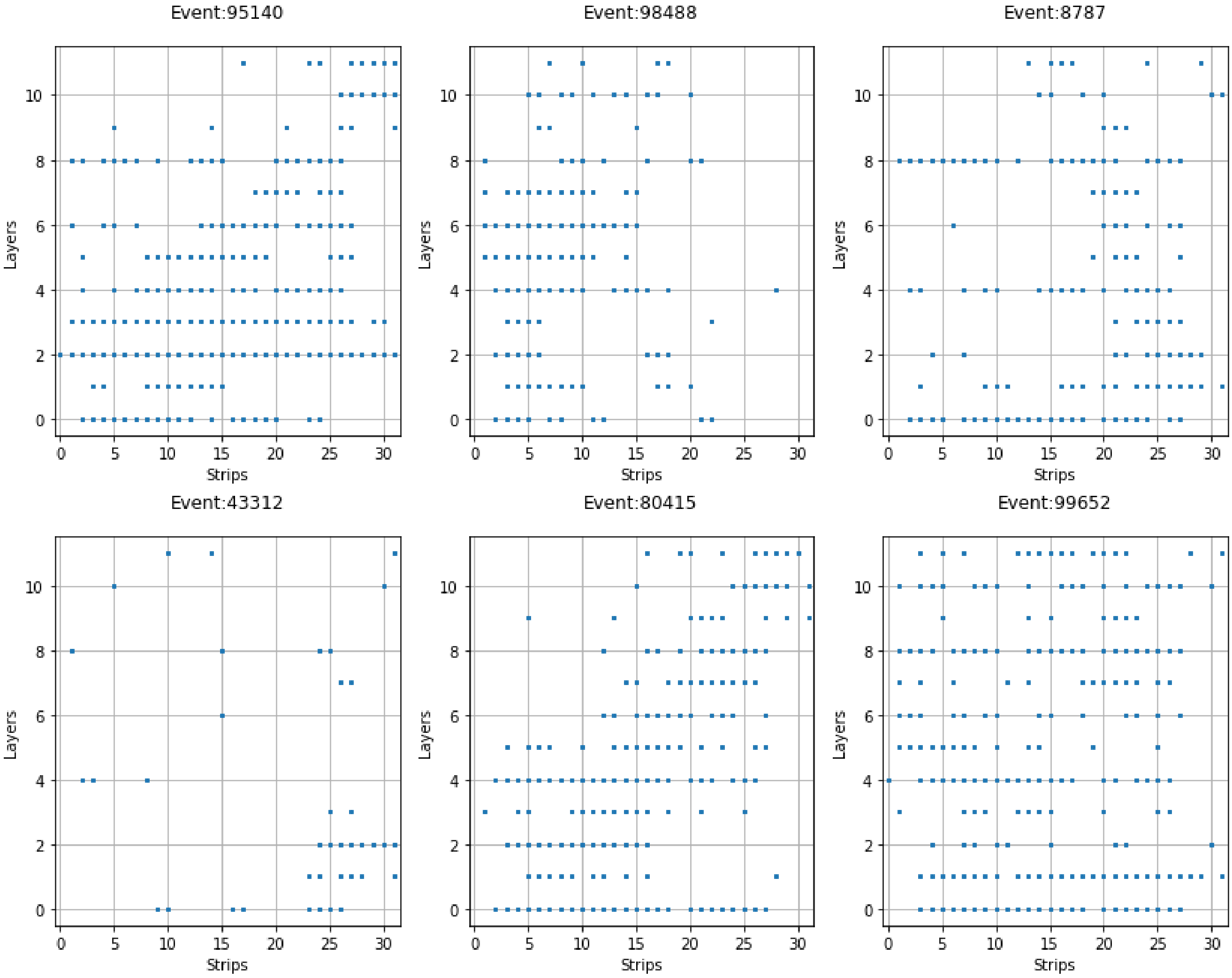

Fig. 7.4 Some correctly predicted noisy events

7.1.4 Single muon events wrongly predicted as noisy events

However, this trained model cannot be declared perfect or ideal model since there has been some wrong predictions also. Some of the single muon events from the dataset which have been wrongly predicted by this trained model as noisy events are shown in figure 7.5.

7.1.5 Multi-muon events wrongly predicted as noisy events

Some of the multi-muon events from the dataset which have been wrongly predicted as noisy events are shown in figures **??**.

7.2 Current status

Till date, about ten lakh real cosmic muon events, as recorded by the TIFR detector prototype stack [Figure 2.1] have been analysed and the same result is given below, well as shown in the figure 7.7

- Total events: 1002700
- Single muon events: 211566 (21.09%)
- Multi muon events: 62080 (6.19%)
- Noisy events: 729055 (72.71%)

7.3 Computational time

One of the important factors behind choosing a reconstruction algorithm is the time it takes to process the output of an event. The training and testing of machine learning were done on google collaboratory, which has a Tesla K80 GPU (12 GB graphics processing memory), 12 GB physical memory, and 360 GB hard disk. All the data sets were generated on a laptop with an Intel(R) Core(TM) i3-7020U CPU @ 2.30GHz with 8 GB physical memory and a 962.4 GB hard-drive with Ubuntu 20.04 LTS operating system. The time taken for simulation, training and testing the

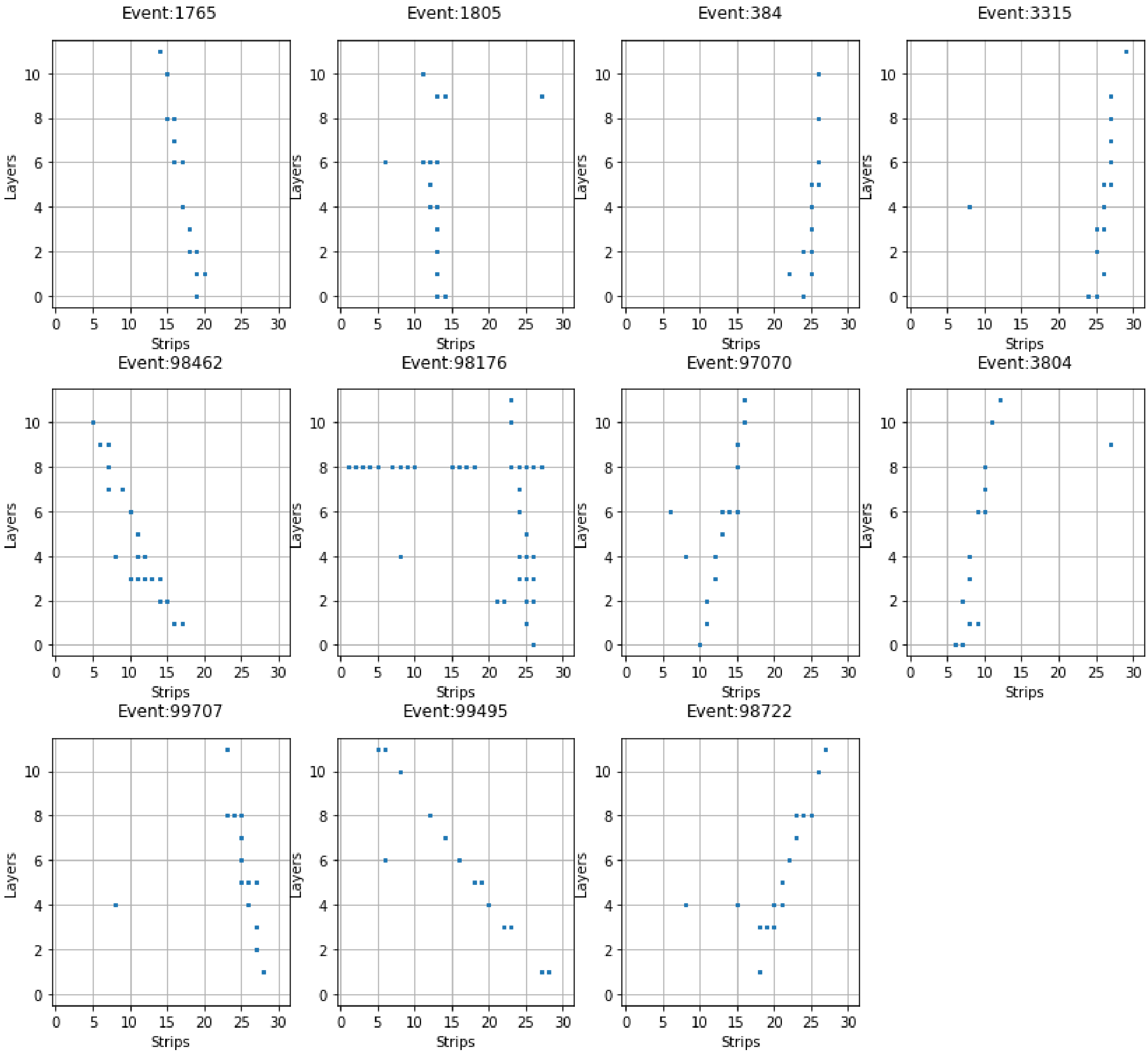

Fig. 7.5 Single muon events wrongly predicted as noisy events

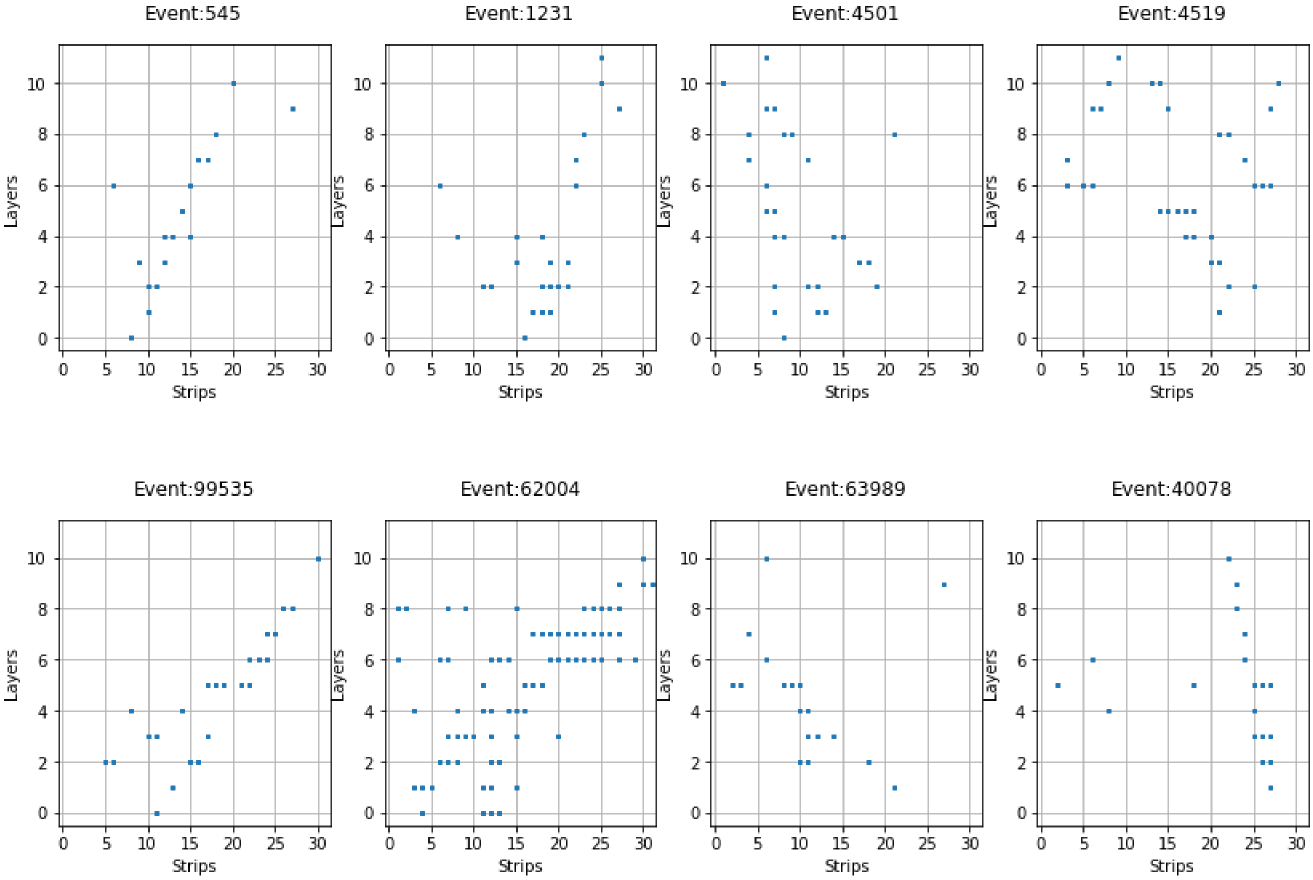

Fig. 7.6 Multi-muon events wrongly predicted as noisy events

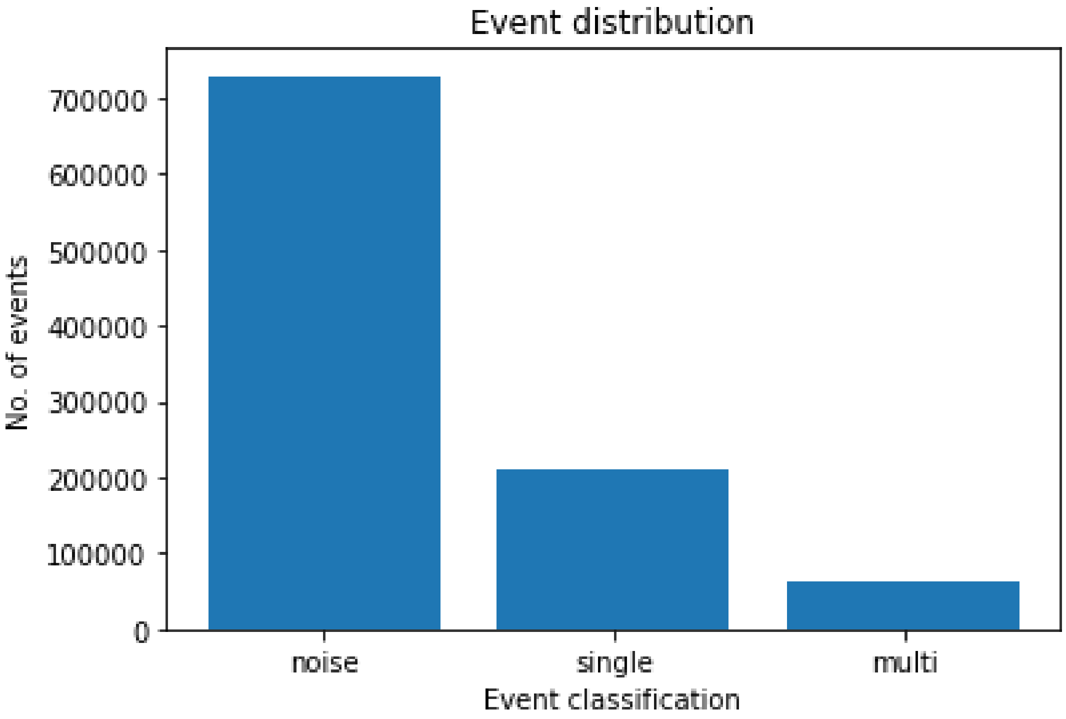

Fig. 7.7 Real event analysis

ML model with data-sets of clean tracks are summarised in table 7.1. In table 7.1, computational time taken for clean tracks are discussed only. The time taken to add η, M_s and M_n to clean tracks are ~ 2.10 minutes, ~ 2.03 minutes, and ~ 2.12 minutes respectively. Time taken for training and testing these noise added data files are almost same as the time taken for training and testing the clean tracks.

Dataset-	Trigger condition	Time taken for data generation	Time taken for training (ML)	Time taken for testing (ML)
Dataset-1	4/12	2 h 08.06 min	2.9 sec	0.08 sec
Dataset-2	5/12	2 h 24.39 min	2.576 sec	0.078 sec
Dataset-3	6/12	2 h 49.52 min	2.55 sec	0.076 sec
Dataset-4	7/12	3 h 8.43 min	2.507 sec	0.075 sec

Table 7.1 Table summarizing the training and testing time for different data-sets. The time mentioned here are all approximated.

The datasets generated for real event analysis, the accuracy calculation of machine learning algorithm on those datasets and real event analysis was done on a computer with an Intel Xeon Gold 6130 (2.10 GHz/16 cores/22 MB/125 W) with 256 GB physical memory and a 1536 GB hard-drive with Cent OS operating system. The time required is specified below:

- **Single muon dataframe (100000 events):** 16 h 51 m
- **Multi muon dataframe (100000 events):** 2 d 1 h 28 m
- **Noisy dataframe (100000 events):** 16 h 30 m
- **Accuracy calculation:** 10 h 32 m

Chapter 8

Conclusions and Future Scope

In this study, we studied the possibility of applying a machine learning-based algorithm to estimate the muon-multiplicity in an INO-ICAL prototype detector. Simplified cosmic muon data is simulated for this purpose which is used to test and train the ML model. The efficiency of the machine learning-based algorithm is found to be better than the efficiency of the Straight Line Fit (SLF) algorithm. The efficiency of predicting multi muon events is greater than 90% for trigger 4/12, greater than 70% for trigger 5/12, almost 73% for trigger 6/12, and 77% for trigger 7/12. Moreover, it is the simplicity of the machine learning algorithm which makes it more applicable to this task.

As a baseline study, this current study was done on simulated data sets. It is worth noting that machine learning algorithms are more efficient and less time-consuming in estimating the muon-multiplicity of an event. The current study can be very useful in calculating the rate of multi-muon events in different prototype stacks of INO all over India. A map of zenith angular distribution for multi-muon events can also be prepared using this algorithm with the data collected from the prototype stacks. These studies can provide a comprehensive understanding of the capability of a machine learning algorithm for multi-muon studies.

References

[1] Albertsson, K., Altoe, P., Anderson, D., Anderson, J., Andrews, M., Espinosa, J. P. A., Aurisano, A., Basara, L., Bevan, A., Bhimji, W., Bonacorsi, D., Burkle, B., Calafiura, P., Campanelli, M., Capps, L., Carminati, F., Carrazza, S., fan Chen, Y., Childers, T., Coadou, Y., Coniavitis, E., Cranmer, K., David, C., Davis, D., Simone, A. D., Duarte, J., Erdmann, M., Eschle, J., Farbin, A., Feickert, M., Castro, N. F., Fitzpatrick, C., Floris, M., Forti, A., Garra-Tico, J., Gemmler, J., Girone, M., Glaysher, P., Gleyzer, S., Gligorov, V., Golling, T., Graw, J., Gray, L., Greenwood, D., Hacker, T., Harvey, J., Hegner, B., Heinrich, L., Heintz, U., Hooberman, B., Junggeburth, J., Kagan, M., Kane, M., Kanishchev, K., Karpiński, P., Kassabov, Z., Kaul, G., Kcira, D., Keck, T., Klimentov, A., Kowalkowski, J., Kreczko, L., Kurepin, A., Kutschke, R., Kuznetsov, V., Köhler, N., Lakomov, I., Lannon, K., Lassnig, M., Limosani, A., Louppe, G., Mangu, A., Mato, P., Meenakshi, N., Meinhard, H., Menasce, D., Moneta, L., Moortgat, S., Neubauer, M., Newman, H., Otten, S., Pabst, H., Paganini, M., Paulini, M., Perdue, G., Perez, U., Picazio, A., Pivarski, J., Prosper, H., Psihas, F., Radovic, A., Reece, R., Rinkevicius, A., Rodrigues, E., Rorie, J., Rousseau, D., Sauers, A., Schramm, S., Schwartzman, A., Severini, H., Seyfert, P., Siroky, F., Skazytkin, K., Sokoloff, M., Stewart, G., Stienen, B., Stockdale, I., Strong, G., Sun, W., Thais, S., Tomko, K., Upfal, E., Usai, E., Ustyuzhanin, A., Vala, M., Vasel, J., Vallecorsa, S., Verzetti, M., Vilasís-Cardona, X., Vlimant, J.-R., Vukotic, I., Wang, S.-J., Watts, G., Williams, M., Wu, W., Wunsch, S., Yang, K., and Zapata, O. (2019). Machine learning in high energy physics community white paper.

[2] Athar, M. S. et al. (2006). India-based Neutrino Observatory: Project Report. Volume I.

[3] Berger, C. et al. (1989). Experimental Study of Muon Bundles Observed in the Frejus Detector. *Phys. Rev. D*, 40:2163.

[4] Bhuyan, M., Chandratre, V., Dasgupta, S., Datar, V., Kalmani, S., Lahamge, S., Mondal, N., Nagaraj, P., Pal, S., Rao, S., et al. (2012). Vme-based data acquisition system for the india-based neutrino observatory prototype detector. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 661:S73–S76.

[5] Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. pages 785–794.

[6] Gillies, E. (2018). Comet phase-i track reconstruction using machine learning and computer vision.

[7] India-based Neutrino Observatory website (2020). India-based neutrino observatory. [Online; accessed April 09, 2020].

[8] Kajita, T. (2016). Nobel lecture: Discovery of atmospheric neutrino oscillations. *Reviews of Modern Physics*, 88(3):030501.

[9] Krishnaswamy, M., Menon, M. G. K., Narasimham, V., Hinotani, K., Ito, N., Miyake, S., Osborne, J., Parsons, A., and Wolfendale, A. W. (1971). The kolar gold fields neutrino experiment i. the interactions of cosmic ray neutrinos. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 323(1555):489–509.

[10] Kumar, A., Kumar, A. V., Jash, A., Mohanty, A. K., Chacko, A., Ajmi, A., Ghosal, A., Khatun, A., Raychaudhuri, A., Dighe, A., et al. (2017). Invited review: Physics potential of the ical detector at the india-based neutrino observatory (ino). *Pramana*, 88(5):79.

[11] McDonald, A. B., Klein, J. R., and L. Wark, D. (2003). Solving the solar neutrino problem. *Scientific American*, 288(4):40–49.

[12] Pal, S. (2014). Development of the ino-ical detector and its physics potential.

[13] Pal, S., Acharya, B., Majumder, G., Mondal, N., Samuel, D., and Satyanarayana, B. (2012a). Measurement of integrated flux of cosmic ray muons at sea level using the ino-ical prototype detector. *Journal of Cosmology and Astroparticle Physics*, 2012(07):033.

[14] Pal, S., Majumder, G., Mondal, N., Samuel, D., and Satyanarayana, B. (2012b). Angular distribution of cosmic muons using ino–ical prototype detector at tifr. *Pramana*, 79(5):1267–1270.

[15] Pauli, W. (1930). Open letter to the group of radioactive people at the gauverein meeting in tübingen. *Translation by Kurt Riesselmann),(December, 4, 1930).*

[16] Raschka, S. and Mirjalili, V. (2019). *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2*. Packt Publishing Ltd.

[17] Reines, F. and Cowan Jr, C. (1953). Detection of the free neutrino. *Physical Review*, 92(3):830.

[18] Samuel, D. and Suresh, K. (2018). Artificial neural networks-based track fitting of cosmic muons through stacked resistive plate chambers. *Journal of Instrumentation*, 13(10):P10035.

[19] Satyanarayana, B. (2009). Design and characterisation studies of resistive plate chambers. *Diss. PhD thesis, Department of Physics, IIT Bombay, PHY-PHD-10-701*.

[20] Shtejer Diaz, K. (2016). Study of muon bundles from extensive air showers with the alice detector at cern lhc. *Journal of Physics: Conference Series*, 718:052039.