Application of Machine Learning Algorithms for Profile Reconstruction of IPM

Submitted by

Alok Anil Jadhav

(2014BEC027)

Department of Electronics and Telecommunication Engineering

Shri Guru Gobind Singhji Institute of Engineering and Technology,

Nanded- 431606, India

Guided by

Dr. A.V. Nandedkar And Dr. Rahul Singh

A thesis submitted for the degree of

Bachelor of Technology

2017-18

This thesis is dedicated to
All the Scientists and the Researchers of GSI, FAIR and CERN
for their lifelong hard-work for the betterment of the humanity.
This book was set in the LaTeXprogramming language by the author.

Acknowledgements

This project would not be possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to Dr. A.V. Nandedkar (Associate Professor, SGGSIET, Nanded, India) and Dr.Rahul Singh (Scientist, GSI/FAIR Labs, Darmstadt, Germany) for their guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project. I would like to express my gratitude towards Dr. A.B. Gonde (HOD, Associate Professor, SGGSIET, Nanded, India) and Dr.Jorn Knoll(Scientist, GSI/FAIR Labs, Darmstadt, Germany) for supporting this project. I would also like to thank my parents as well as members of GSI Lab and SGGSIET for their kind cooperation and encouragement which help me in the completion of this project.

My thanks and appreciations also go to my colleague for developing the nurturing environment around and people who have willingly helped me out with their abilities.

Contents

List of Figures

List of Tables

Abstract

A particle accelerator is a machine that uses electromagnetic fields to propel charged particles to nearly light speed and to contain them in well-defined beams. [2] Beams of high-energy particles are useful for fundamental and applied research in the sciences, and also in many technical and industrial fields unrelated to fundamental research. It has been estimated that there are approximately 30,000 accelerators worldwide [3]. IPM is used to measure transverse beam properties of accelerated beam by accelerators. Feed-back from IPM is provided to control system so that it can control the beam measurements. Measured IPM profiles can be significantly distorted due to displacement of residual ions or electrons by interaction with beam fields for high brightness or high energy beams [1, 4, 5, 6]. It is thus difficult to deduce the characteristics of the actual beam profile from the measurements. Machine learning is a field of computer science that gives computers the ability to learn without being explicitly programmed. In this field we can train machines to excel in well defined tasks by using different learning methods. Machine Learning regression can be used to predict original profile by using different regression algorithms like linear regression, ridge regression, kernel ridge regression, support vector machines [29]. In this project different Machine Learning Regression Algorithms are applied to reconstruct the actual beam profile from the measurement data. These Regression algorithms are trained using Virtual-IPM simulation program [7] developed under the IPMSim collaboration [13]. The results from different algorithms are presented in this contribution.

Chapter 1

Project Background

1.1 Introduction

Ionization Profile Monitors (IPM) are used for non-destructive transverse beam profile measurements at many accelerator facilities. The principle of operation is the following; the primary beam ionizes the residual gas and the ionized particles (ions or electrons) are extracted via electric fields and sometimes in conjunction with magnetic fields to confine the movement of ionized particles in the plane transverse to the electric field. The profile of the extracted particles reflects the transverse profile of the primary beam with the assumption that ionized particles are created at rest and the effect of induced fields by the primary beam on ionized particles can be neglected. The choice between ions or electrons for profile reconstruction is based on the requirement for the speed of device operation and potential influence of beam space charge.

Figure 5.1 shows the typical components present in an IPM where both the electric and magnetic fields are utilized to confine the ionized particles [14]. The support electrodes/rods between the top and bottom electrodes are used to reduce the fringe fields and improve field homogeneity. The field homogeneity is important in order to avoid any distortion in the measured profile and therefore static EM simulations for the full geometry are usually performed. IPMs are often used for non-destructive measurements in low-pressure conditions such as storage rings and hence they usually have to be equipped with a high amplification multi-channel plate (MCP) for obtaining sufficient signal to noise ratio. The output of MCPs is connected to data acquisition system directly or via phosphor screens and an optical system. Figure 1.2 shows the image of the horizontal IPM formerly installed at LHC relevant to the discussions in this paper [4]. The dipole magnet has been moved in order to make the IPM chamber visible.

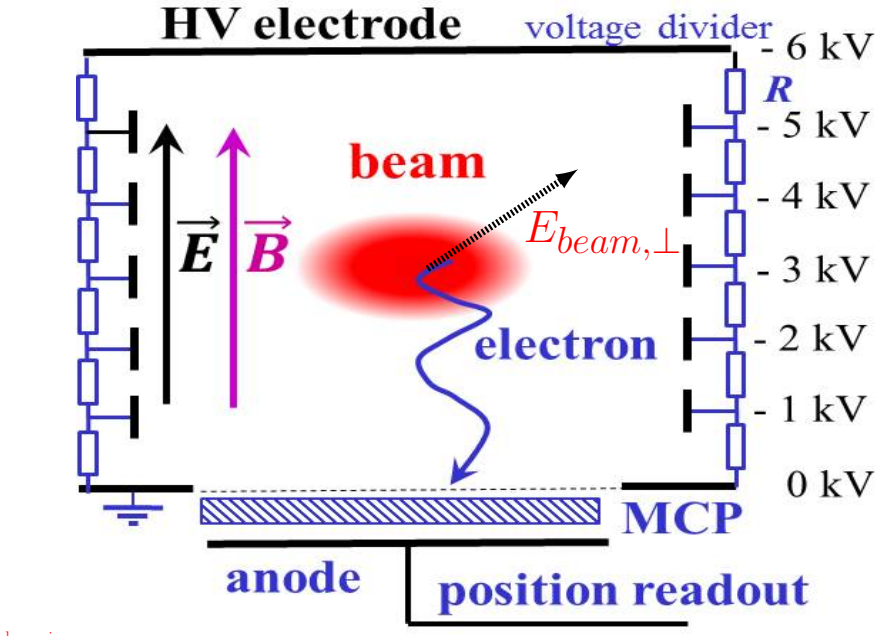

Figure 1.1: Operating principle of the IPM.

In the next section, we will discuss some basics about particle accelerators and distortion in measured IPM profiles due to beam space charge and discuss previous efforts on correcting or reducing the distortion. Following that, the simulation tool and the beam and device parameters used to train the Machine Learning Regression models are discussed. Finally, the Machine Learning Regression parameters, training and validation are presented and the results are summarized.

1.2 Motivation for this project

1. Many accelerator facilities (CERN-Switzerland, GSI/FAIR-Germany, DESY Germany, J-PARC-Japan) are trying to solve the problem of IPM profile reconstruction by using mathematical modeling of the system but error percentage for the reconstruction is high [13]. Due to distortion in measured profiles, the properties of accelerated beams can't be measured efficiently.

2. The error percentage in reconstructed profile by using current techniques of reconstruction is high which requires the approach of theoretical physics and mathematics for reconstruction of distorted profile. [6] [7]

3. To extend earlier work conducted by researchers at GSI/FAIR lab, Germany. [16] [31] and To make the results implementable at the end of the project in live accelerator facilities.

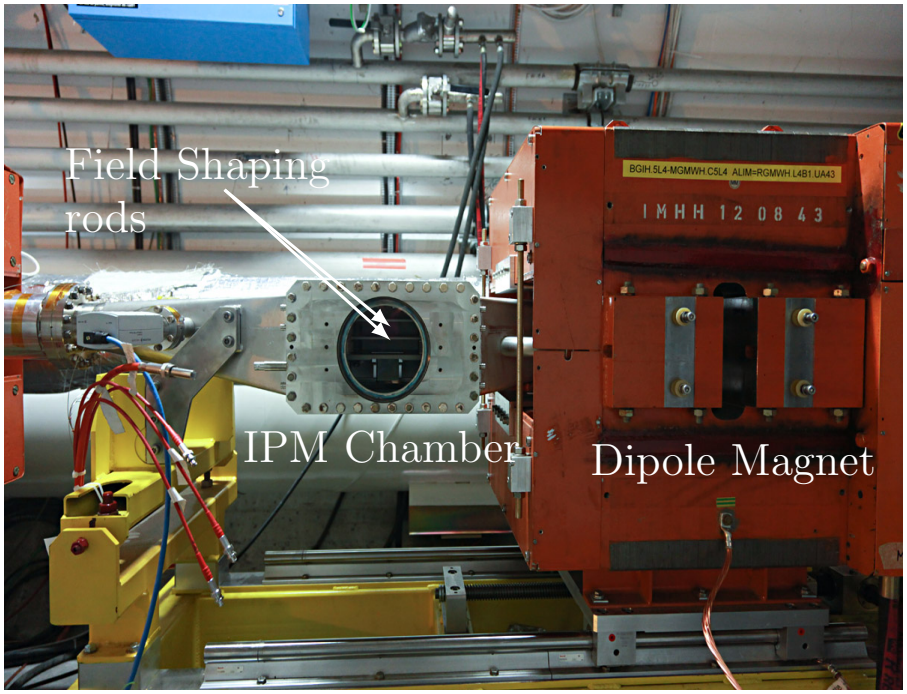

Figure 1.2: IPM installation at LHC. The dipole magnet (orange) has been shifted, revealing the IPM chamber.

1.3 Objectives

Goal of the project is to build a system for predicting or reconstructing the original profile of IPM. This requires thorough study of regression algorithms and factors affecting them. The effective algorithms will be implemented in real accelerator facilities. Following are the proposed objectives:

1] To find factors affecting profile of IPM.

2] To reconstruct distorted profile by using regression algorithms.

3] To find out factors affecting error percentage of different regression algorithms while reconstruction of profile.

1.4 Executive Summary

Summary of our project can be depicted as shown in given Figure 1.3:

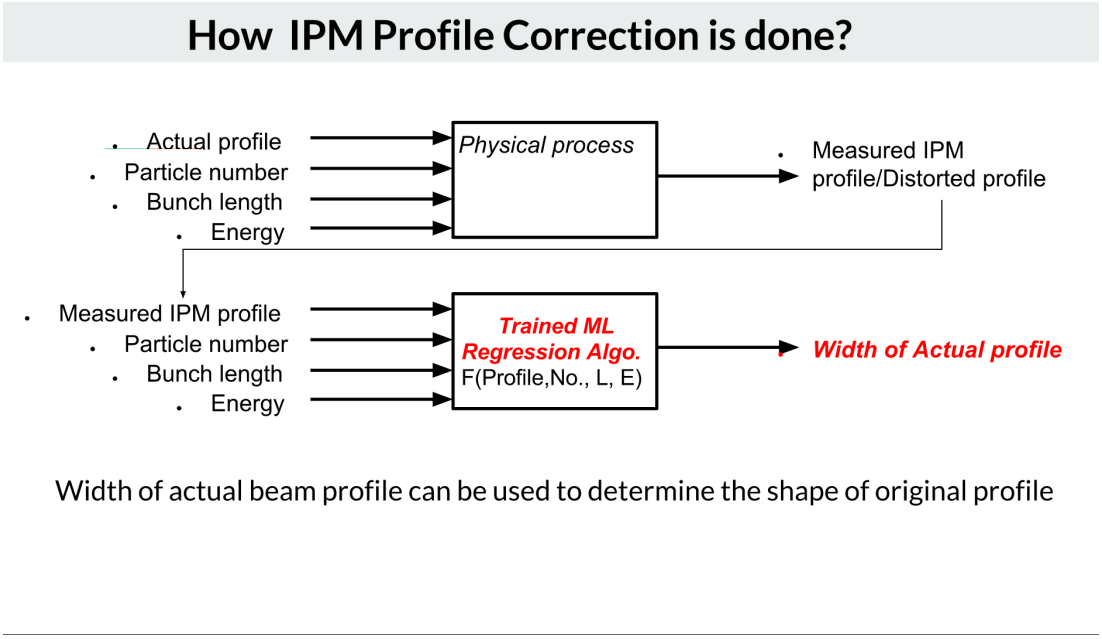

Figure 1.3: Method for profile reconstruction problem

Chapter 2

Literature Survey

Accelerator Physics and IPM Research:

As described by J. Egberts et al. in 2012 in his PhD thesis, In its simplest design, a linear accelerator consists of a source, where the particles are created, an accelerating structure, where these particles are accelerated to design energy,and some kind of target where the beam is finally put to use. For a real accelerator, one requires various components in addition, like focusing and shaping elements, vacuum pumps, valves, diagnostics, of course, and many more. Each single component of the accelerator has to be designed and optimized in order to obtain the expected beam characteristics at given positions. Beam characteristics required for the accelerator design include not only rather obvious quantities like beam current or transverse position and size, but also quantities like its longitudinal size, angular dispersion, energy spread , or the number of lost particles in the accelerator that have to be known with astonishing precision [3].

M. Sapinski et al., in 2012 published paper which shows that, Ionization Profile Monitors (IPM) on LHC are configured to measure electrons produced in ionization of Neon gas, injected into LHC vacuum chamber. The beam passes between two ceramic electrodes with difference of potentials of 4 kV, over a distance of 85 mm. This potential brings the electrons to Multi-Channel Plate (MCP, from Photonis), where the signal is amplified. A phosphor screen is located 2 mm behind the MCP. It is deposited on a right-angle prism, which is the only optical element inside the vacuum chamber [4].

Studies conducted by D. Vilsmeier et al., in 2014 concludes that, Measuring the transverse beam size in the Large Hadron Collider by using Ionization Profile Monitors is a difficult task for energies above injection during the energy ramp from 450 GeV to 6.5 TeV. The beam size decreases from around 1 mm to 200 μm and the brightness of the beam is high enough to destroy the structure of any form of interacting

matter. While the electron trajectories are confined by an external electro-magnetic field which forces the electrons accordingly on helix paths with certain gyroradii, this gyration is heavily increased under the influence of the electric field of the beam. Smaller beam sizes, which go hand in hand with increased bunch electric fields, lead to larger gyroradii of the ionized electrons, which results in strongly distorted profiles. In addition, this distortion becomes more visible for smaller beam sizes as the extent of gyration grows compared to the actual beam size. Depending on the initial momentum distribution of the electrons, emerging from the ionization process with the highly relativistic beam, the profile distortion is affected significantly. In order to be able to perform reliable investigations into the effects of space charge a good knowledge of such initial momentum distributions is essential. The theoretical calculation of electron initial momenta will be discussed in order to obtain reliable results from the simulation of the electron movement within the ionization chamber which are used to investigate the effect of space charge on the registered profiles [6].

Machine Learning Application:

As per Arthur Samuel et al., Machine Learning provides machines ability to learn without being explicitly programmed [20]. It also gives machine ability to learn from data and to predict the behavior of systems or to make certain decisions. The ability to find the underlying process model and its features from data are referred to as machine learning. This can be used to predict the system behavior or make decisions. In this method, computers are trained to recognize patterns in the vast amount of data. The field is evolved primarily from Pattern Recognition and Artificial Intelligence. Though ML algorithms have a variety of applications such regression, classification, dimensionality reduction, clustering, density estimation, we focus on regression/prediction in our context. In general, there are three types of machine learning algorithms: supervised learning, unsupervised learning, reinforcement learning. Problem like Reconstruction of distorted beam profile can be solved by using machine learning algorithms [28].

Chapter 3

Particle Accelerators and IPM

3.1 Particle Accelerators

3.1.1 Introduction To Particle Accelerators

A particle accelerator is a machine that utilizes electromagnetic fields to propel charged particles to proximately light speed and to contain them in well-defined beams [8]. Large accelerators are utilized in particle physics as colliders (e.g., the LHC at CERN, J-PARC in Japan, SIS-18, SIS-100 and UNILAC at GSI/FAIR Germany, RHIC at Brookhaven National Laboratory, and Tevatron at Fermilab), or as synchrotron light sources for the study of condensed matter physics.More diminutive particle accelerators are utilized in a wide variety of applications, including particle therapy for oncological purposes, radioisotope production for medical diagnostics, ion implanters for the manufacture of semiconductors, and accelerator mass spectrometers for measurements of rare isotopes such as radiocarbon. There are currently more than 30,000 accelerators in operation around the world [9].

There are two basic classes of accelerators: electrostatic and electrodynamic (or electromagnetic) accelerators [10]. Electrostatic accelerators use static electric fields to accelerate particles. The most common types are the CockcroftWalton generator and the Van de Graaff generator. A small-scale example of this class is the cathode ray tube in an ordinary old television set. The achievable kinetic energy for particles in these devices is determined by the accelerating voltage, which is limited by electrical breakdown. Electrodynamic or electromagnetic accelerators, on the other hand, use changing electromagnetic fields (either magnetic induction or oscillating radio frequency fields) to accelerate particles. Since in these types the particles can pass through the same accelerating field multiple times, the output energy is not limited

by the strength of the accelerating field. This class, which was first developed in the 1920s, is the basis for most modern large-scale accelerators.

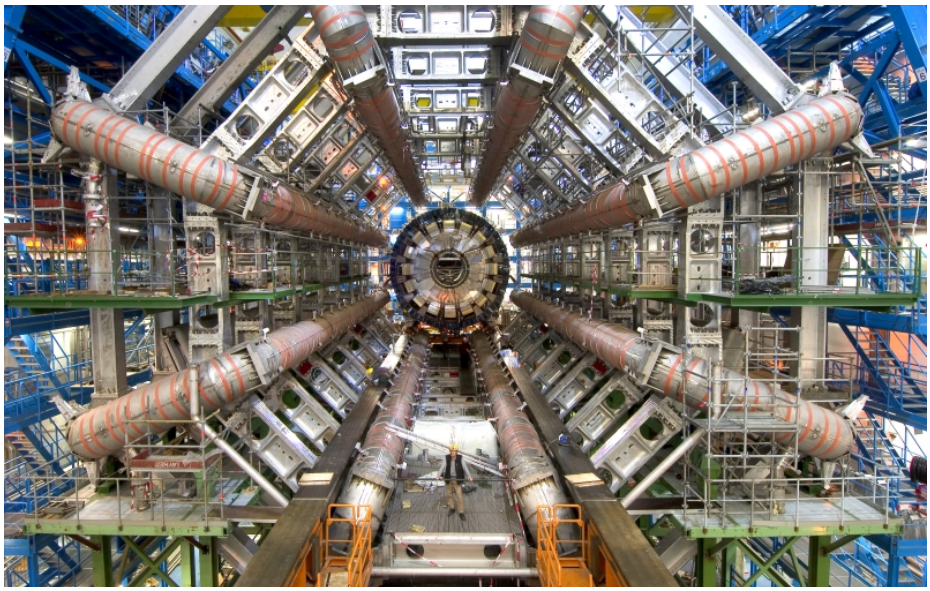

Figure 3.1: Particle Accelerator LHC (large hydron collider) situated at CERN Switzerland

3.1.2 How The Accelerator Works

Accelerators were invented in the 1930s to provide energetic particles to investigate the structure of the atomic nucleus. Since then, they have been used to investigate many aspects of particle physics. Their job is to speed up and increase the energy of a beam of particles by generating electric fields that accelerate the particles, and magnetic fields that steer and focus them. An accelerator comes either in the form of a ring (a circular accelerator), where a beam of particles travels repeatedly round a loop or in a straight line (a linear accelerator), where the particle beam travels from one end to the other. At CERN a number of accelerators are joined together in sequence to reach successively higher energies. The type of particle used depends on the aim of the experiment. The Large Hadron Collider (LHC) accelerates and collides protons, and also heavy lead ions. One might expect the LHC to require a large source of particles, but protons for beams in 27-kilometer ring come from a single bottle of hydrogen gas, replaced only twice per year to ensure that it is running at the correct pressure.

How to accelerate protons: In the first part of the accelerator, an electric field strips hydrogen nuclei (consisting of one proton and one electron) of their electrons.

Electric fields along the accelerator switch from positive to negative at a given frequency, pulling charged particles forwards along the accelerator. CERN engineers control the frequency of the change to ensure the particles accelerate not in a continuous stream, but in closely spaced bunches.

Radiofrequency (RF) cavities: These are specially designed metallic chambers, spaced at intervals along the accelerator are shaped to resonate at specific frequencies, allowing radio waves to interact with passing particle bunches. Each time a beam passes the electric field in an RF cavity, some of the energy from the radio waves is transferred to the particles, nudging them forwards. Its important that the particles do not collide with gas molecules on their journey through the accelerator, so the beam is contained in an ultrahigh vacuum inside a metal pipe the beam pipe.

Various types of a magnet: These serve different functions around a circular accelerator. Dipole magnets, for example, bend the path of a beam of particles that would otherwise travel in a straight line. The more energy a particle has, the greater the magnetic field needed to bend its path. Quadrupole magnets act likes lenses to focus a beam, gathering the particles closer together. Collisions at accelerators can occur either against a fixed target or between two beams of particles.

Particle detectors: These are placed around the collision point to record and reveal the particles that emerge from the collisions.

3.1.3 Uses of The Particle Accelerators

Beams of high-energy particles are useful for fundamental and applied research in the sciences, and also in many technical and industrial fields unrelated to fundamental research. It has been estimated that there are approximately 30,000 accelerators worldwide. Of these, only about 1% are research machines with energies above 1 GeV, while about 44% are for radiotherapy, 41% for ion implantation, 9% for industrial processing and research, and 4% for biomedical and other low-energy research. The bar graph shows the breakdown of the number of industrial accelerators according to their applications. The numbers are based on 2012 statistics available from various sources, including production and sales data published in presentations or market surveys, and data provided by a number of manufacturers [11].

High-energy physics: For the most basic inquiries into the dynamics and structure of matter, space, and time, physicists seek the simplest kinds of interactions at the highest possible energies. These typically entail particle energies of many GeV, and the interactions of the simplest kinds of particles: leptons (e.g. electrons and positrons) and quarks for the matter, or photons and gluons for the field quanta.

Since isolated quarks are experimentally unavailable due to color confinement, the simplest available experiments involve the interactions of, first, leptons with each other, and second, of leptons with nucleons, which are composed of quarks and gluons. To study the collisions of quarks with each other, scientists resort to collisions of nucleons, which at high energy may be usefully considered as essentially 2-body interactions of the quarks and gluons of which they are composed. Thus elementary particle physicists tend to use machines creating beams of electrons, positrons, protons, and antiprotons, interacting with each other or with the simplest nuclei (e.g., hydrogen or deuterium) at the highest possible energies, generally hundreds of GeV or more.

Nuclear physics and isotope production: Nuclear physicists and cosmologists may use beams of bare atomic nuclei, stripped of electrons, to investigate the structure, interactions, and properties of the nuclei themselves, and of condensed matter at extremely high temperatures and densities, such as might have occurred in the first moments of the Big Bang. These investigations often involve collisions of heavy nuclei of atoms like iron or gold at energies of several GeV per nucleon. The largest such particle accelerator is the Relativistic Heavy Ion Collider (RHIC) at Brookhaven National Laboratory. Particle accelerators can also produce proton beams, which can produce proton-rich medical or research isotopes as opposed to the neutron-rich ones made in fission reactors; however, recent work has shown how to make 99Mo, usually made in reactors, by accelerating isotopes of hydrogen, [12] although this method still requires a reactor to produce tritium. An example of this type of machine is LANSCE at Los Alamos.

Low-energy machines and particle therapy: Everyday examples of particle accelerators are cathode ray tubes found in television sets and X-ray generators. These low-energy accelerators use a single pair of electrodes with a DC voltage of a few thousand volts between them. In an X-ray generator, the target itself is one of the electrodes. A low-energy particle accelerator called an ion implanter is used in the manufacture of integrated circuits. At lower energies, beams of accelerated nuclei are also used in medicine as particle therapy, for the treatment of cancer. DC accelerator types capable of accelerating particles to speeds sufficient to cause nuclear reactions are Cockcroft-Walton generators or voltage multipliers, which convert AC to high voltage DC, or Van de Graaff generators that use static electricity carried by belts.

Figure 3.2: Medical Linear Accelerator used for cancer therapy at Stanford medical school USA

3.2 Space Charge Effects On IPM Profile

IPM profile distortion due to beam fields depends on a variety of parameters such as device geometry, beam properties, extracted particle types (ions or electrons) and if IPM uses only electric field or also a magnetic field. Initial IPM developments were focused on devices utilizing only electric fields. The distortion of IPM profiles has long been observed and the first attempt to make a simulation-based correction was by Thern [1] at AGS. Calculation of actual beam profile width from the measured profile as a function of applied electric field and bunch population for coasting and bunched beams was found. Further modifications were made to the Thern analytical model and were experimentally applied at FNAL [2]. Such analytical models made assumptions concerning the beam profile shape and predictions diverged for dense beams. A numerical approach was attempted at CEA Saclay during the development of LIPAc IPMs, where very high space charge is expected [3]. In that correction procedure, first simulations were used to map the actual profile with the measured profile and the mapping was stored in matrices for a range of actual beam profiles modeled as generalized Gaussian distribution and for a range of beam currents. A fast iterative procedure was implemented to utilize these matrices for profile reconstruction experimentally. Practical limitations of this approach was that error in reconstructed profile was dependent on the size or coarseness of the simulation grid, and further reduction of grid size increased the number of reconstruction matrices significantly due to the curse of dimensionality. An alternative approach for dealing with profile

distortion is to use magnetic fields to confine the generated ionized particles around the point of generation. However, in addition to being expensive, required magnetic field strengths are prohibitory for extremely dense beams as discussed below for our target case. First major distortion for an IPM with magnetic fields was seen for LHC IPMs at high energies, where the beam profile was significantly broader compared to wire scanner measurements [4]. First solution envisaged to solve the issue was to raise the magnetic fields in the IPM to $1T$[5]. This field strength should be viewed in perspective of the strength of LHC main dipoles which is $8.5T$ at top energies and is considered impractical.

Recently reliable IPM simulation tools have been developed as a joint effort between several labs [13]. Availability of reliable description of IPM system including the profile distorting effects such as space charge and initial velocity distribution of ionized particles transforms the problem of IPM profile correction into a "supervised learning" problem. In a supervised learning problem, the input and output of an unknown system are provided and the system is approximated by a variety of machine learning algorithms [22]. We have chosen the Machine Learning Regression Algorithms to reconstruct actual beam profile from the measured distorted profile. We have also simplified the problem by assuming Gaussian primary beams for first tests presented here, however, these are not limitations of the method itself and can be extended to arbitrary profiles.

3.3 Virtual-IPM and Simulation Data Details

Table 3.1: Parameters used for the simulation. σ_x, σ_y, σ_l and the bunch population have been varied within the specified intervals.

Particle type	Protons
Energy/u	6.5TeV
Bunch population N_p	1.1 to 1.7×10^{11}
Bunch length σ_l (4σ)	0.9ns to 1.2ns
Bunch width σ_x (1σ)	0.29mm to 0.37mm
Bunch height σ_y (1σ)	0.4mm to 0.6mm
Electrode distance	85mm
Applied voltage	4kV
Magnetic field	0.2T

The Virtual-IPM simulation program has been used in order to generate beam profiles for LHC parameters. Table 3.1 shows the parameters which have been used

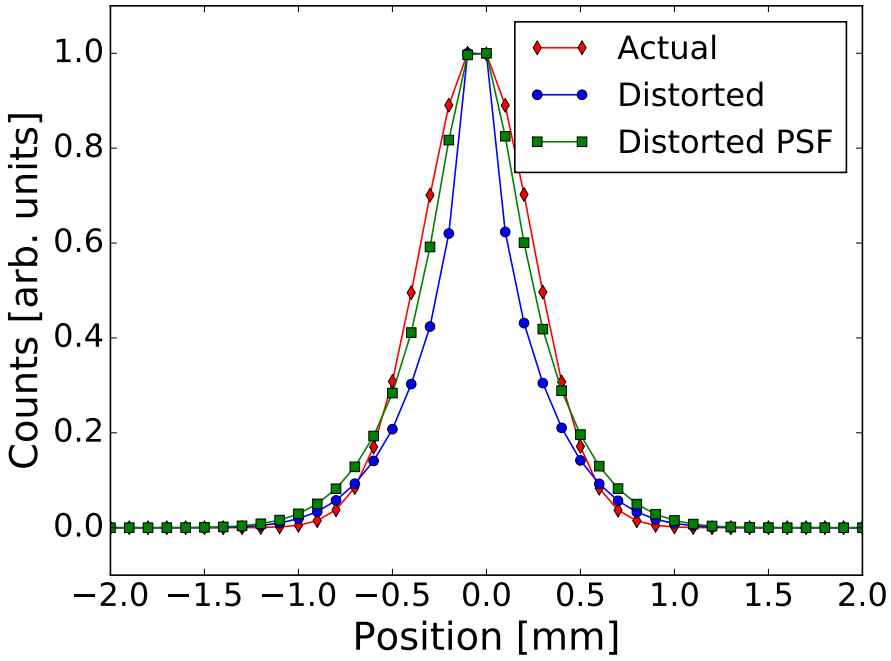

Figure 3.3: Simulation of profile distortion due to space charge using Virtual IPM.

for the simulations. σ_x, σ_y, σ_l and the bunch population N_p have been varied to cover the relevant operational region. The simulation used a Gaussian bunch shape and an analytical solution for the electric field of a Gaussian bunch in two dimensions, neglecting the longitudinal field component. This is justified because the bunch is highly relativistic and is longitudinal size is significantly larger than transverse size. The initial velocities of electrons were sampled from the Voitkiv double differential cross section [16] for a Hydrogen target. Each case simulated 1×10^6 particles whose final positions were grouped into profiles with 100μm bin size.

One million particles were used in each simulation providing rather smooth profiles. The bin size is 1mm and Gaussian point spread function with $\sigma = 0.125$ was used to represent the effect of optical acquisition system in the LHC IPM system. Figure 3.3 shows the actual primary beam profile, the distorted profile due to space charge and the profile read at the end of the acquisition system after application of point spread function of the optics. The input parameters to simulation were N_p =1.7×10^{11}, $\sigma_l = 0.9ns$, $\sigma_x = 0.29mm$, and $\sigma_y = 0.4mm$ There is a 20 increase in the second central moment of the measured profile (with PSF) with respect to actual (initial) profile.

Chapter 4

Machine Learning Regression

4.1 Introduction to ML

Machine learning algorithms have a variety of applications such regression, classification, dimensionality reduction, clustering, density estimation, we focus on regression/prediction in our context. In general, there are three types of machine learning algorithms: supervised learning, unsupervised learning, reinforcement learning [28]. The choice of ML algorithm is problem specific, and typical deciding factors are bias-variance tradeoff, training time, and dimensionality of the problem (i.e.the number of inputs and outputs), and any prior information about the features of the underlying model.

4.2 How To Select Algorithm For Specific Problem

We need to select the algorithm from the specific group(supervised learning, unsupervised learning or reinforcement learning) depending on many factors(only a few of them are mentioned here) like training time, accuracy, number of features required.

Accuracy: Getting the most accurate answer possible isn't always necessary. Sometimes an approximation is adequate, depending on what we want to use it for. If that's the case, we may be able to cut our processing time dramatically by sticking with more approximate methods. Another advantage of more approximate methods is that they naturally tend to avoid overfitting. Like considering that data behaves as Gaussian distribution will make data interpretable.

Training time: The number of minutes or hours necessary to train a model varies a great deal between algorithms. Training time is often closely tied to accuracyone typically accompanies the other.

The number of parameters: Parameters are the knobs a data scientist gets to turn when setting up an algorithm. They are numbers that affect the algorithm's behavior, such as error tolerance or the number of iterations, or options between variants of how the algorithm behaves. The training time and accuracy of the algorithm can sometimes be quite sensitive to getting just the right settings. Typically, algorithms with large numbers parameters require the most trial and error to find a good combination.

The number of features: In machine learning and pattern recognition, a feature is an individual measurable property or characteristic of a phenomenon being observed [23]. For certain types of data, the number of features can be very large compared to the number of data points. This is often the case with genetics or textual data. A large number of features can slow down some learning algorithms, making training time much longer. Support Vector Machines are particularly well suited to this case. [28]

4.3 Types and working principle behind Machine Learning Algorithms

Each ML regression algorithms involve three important functions:

"Machine Learning = Representation + Evaluation + Optimization"

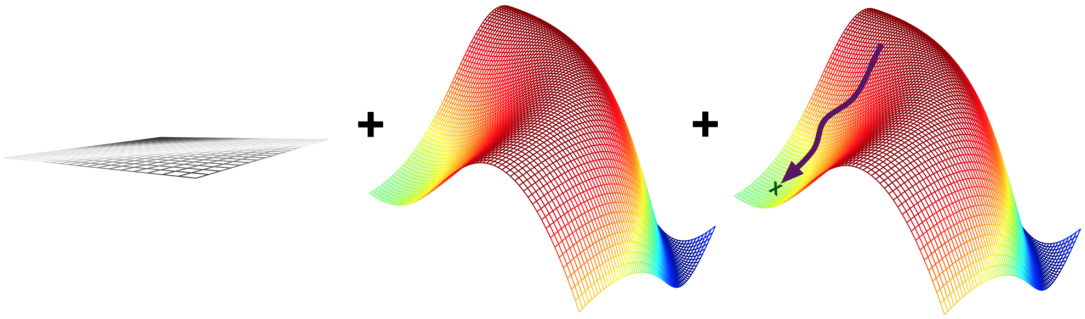

Figure 4.1: Machine Learning algorithms as the combination of landscape, preference, and strategy

A) Mathematical learning model (Representation): It represents an approximation of the underlying function between inputs and outputs which is later used for predictions of the unknown input dataset. Representation is basically the space of allowed models (the hypothesis space) but also takes into account the fact that we are expressing models in some formal language that may encode some models more easily than others (even within that possible set). This is akin to the landscape of possible

models, the playing field allowed by a given representation. For example, 3-layer feed-forward neural networks (or computational graphs) form one type of representation, while support vector machines with RBF kernels form another.

B) Loss function/Error function (Evaluation): Function indicating the $L : y \times y \rightarrow \mathbb{R}_+$ penalty for an incorrect prediction [35]. It is used for finding error percentage between predicted output and true output. Evaluation is essentially how you judge or prefer one model vs. another; its what you might have seen as a utility function, loss function, scoring function, or fitness function in other contexts. Think of this as the height of the landscape for each given model, with lower areas being preferable/desirable than higher areas (without loss of generality). Mean squared error (of a models output vs. the data output) or likelihood (the estimated probability of a model given the observed data) are examples of different evaluation functions that will imply somewhat different heights at each point on a single landscape.

C) Optimization Function(Optimization): it is used to search optimal parameters for mathematical learner model. In most algorithms "gradient descent method" is used as optimization function. This is how you search the space of represented models to obtain better evaluations. This is the way you expect to traverse the landscape to find the promised land of ideal models; the strategy of getting to where you want to go. Stochastic gradient descent and genetic algorithms are two (very) different ways of optimizing a model class. Note that once you provide a trained model, its quite possible that you may no longer be able to recover exactly how it was optimized. Just because I can see you standing in the pit doesnt mean I know how you got there.

There are three main categories of machine learning: supervised learning, unsupervised learning, and reinforcement learning.

A) In supervised learning, each data point is labeled or associated with a category or value of interest. An example of a categorical label is assigning an image as either a cat or a dog. An example of a value label is the sale price associated with a used car. The goal of supervised learning is to study many labeled examples like these and then to be able to make predictions about future data points. For example, identifying new photos with the correct animal or assigning accurate sale prices to other used cars. This is a popular and useful type of machine learning.

B) In unsupervised learning, data points have no labels associated with them. Instead, the goal of an unsupervised learning algorithm is to organize the data in some way or to describe its structure. This can mean grouping it into clusters, as K-means does, or finding different ways of looking at complex data so that it appears simpler.

C) In reinforcement learning, the algorithm gets to choose an action in response to each data point. It is a common approach in robotics, where the set of sensor readings at one point in time is a data point, and the algorithm must choose the robots next action. It's also a natural fit for the Internet of Things applications. The learning algorithm also receives a reward signal a short time later, indicating how good the decision was. Based on this, the algorithm modifies its strategy in order to achieve the highest reward.

4.4 Regression Algorithms using Supervised Learning

In our problem statement, the data provided by IPM devices is labeled So we can use supervised learning approach to solve the problem. Four typical ML regression methods with increasing level of complexity are applied to the problem as discussed below:

A) Linear Regression:

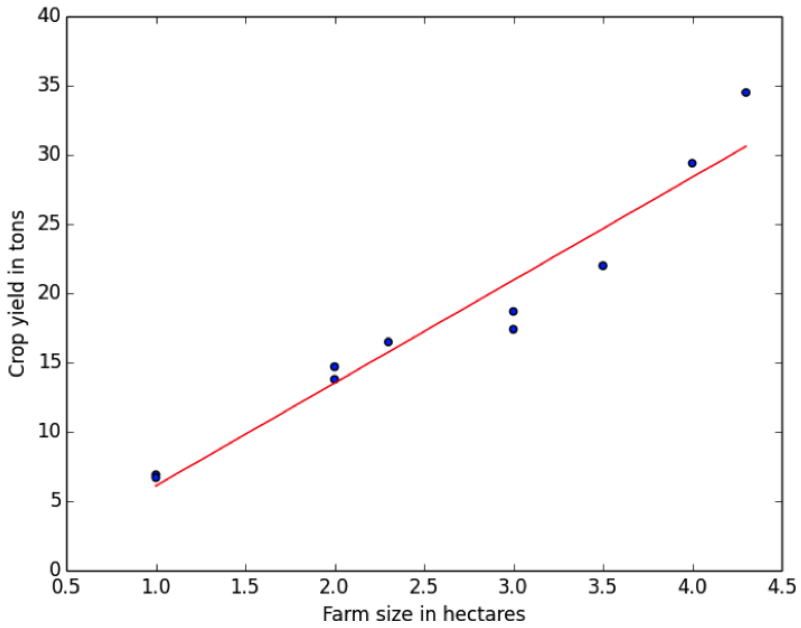

Figure 4.2: Two-dimensional plot of linear regression technique which is trying to fit data points in linear relationship with each other.

It is a linear approach to modeling the relationship between a scalar dependent variable y and one or more explanatory variables denoted X. The model of Mathe-

matical learning model of linear regression can be represented by,

$$y_{pred} = WX + b \tag{4.1}$$

where W is an array of coefficients and X(X_1,X_2. X_n) is a $M \times N$ matrix where M is the number of data sets, N is the number of features of each data set. b is bias, predicted output is y_{pred}.Predicted output y_{pred} is an array of length M. whereas the equation of loss function of linear regression is [26],

$$min \sum (y_{real} - y_{pred})^2 \tag{4.2}$$

where y_{real} is true output values for specific input value X. There are some shortcomings of linear regression with respect to co-linearity of input variables

Predictive ability: Linear regression has a low bias (zero bias) but suffers from the high variance. So it may be worth sacrificing some bias to achieve a lower variance. Interpretative ability: with a large number of predictors, it can be helpful to identify a smaller subset of important variables.Linear regression does not do this [24]. overfitting: If a number of features of the element of X is comparable to the number of total elements of X then linear regression tends towards overfitting [25] [26].

B) Ridge Regression: To avoid mentioned shortcomings of linear regression, a regularization term is introduced in Ridge regression model [26]. Ridge regression uses similar Mathematical learning model as that of linear regression with a regularization/penalty term for the magnitude of estimated weights. The error function of linear regression is:

$$min \sum (y_{real} - y_{pred})^2 + \lambda(W)^2 \tag{4.3}$$

where regularization term is $\lambda(W)^2$. Which means that weight of coefficient should be as minimum as possible. If an input attribute has a small effect on improving the error function it is shut down by the penalty term. The inclusion of a shrinkage penalty is often referred to as regularization. Ridge regression behaves better than that of linear regression. For problems related to the number of feature number of test data points, Kernel ridge regression or SVM regression is used [28]. Kernel Ridge Regression(KRR) and Support Vector Machines Regression(SVM Regression) use one method to transform lower dimensional input space into higher dimensional feature space.

C)Kernel Trick:

Kernels are used to transform feature space into higher Dimensional feature space and learn a linear classifier/model in that transformed space. We use kernel functions

to do this transformation implicitly. While using SVM/Kernel Ridge for regression or classification, it is required to convert a lower Dimensional data into higher Dimensional data. But this process of converting lower Dimensional data into higher Dimensional data is computationally expensive and very complicated. The final output of such a system is just a single value (because we are comparing two elements/the similarity between two elements that will give us a single value). So instead of converting all this lower dimensional feature space into a higher dimension, kernel trick calculates the dot product of all features and calculate a kernel function (which is already defined) in lower feature space itself. Dot product gives us the similarity between two vectors. So in general kernel is giving output as the similarity between two vectors [27]. More the similarity more the value of dot product. When finally kernel function is calculated it is possible to learn a linear function in that space. we transform the training data into a higher dimensional feature space via a mapping function $\Phi(.)$ [21] such that,

$$X \rightarrow \Phi(X)$$

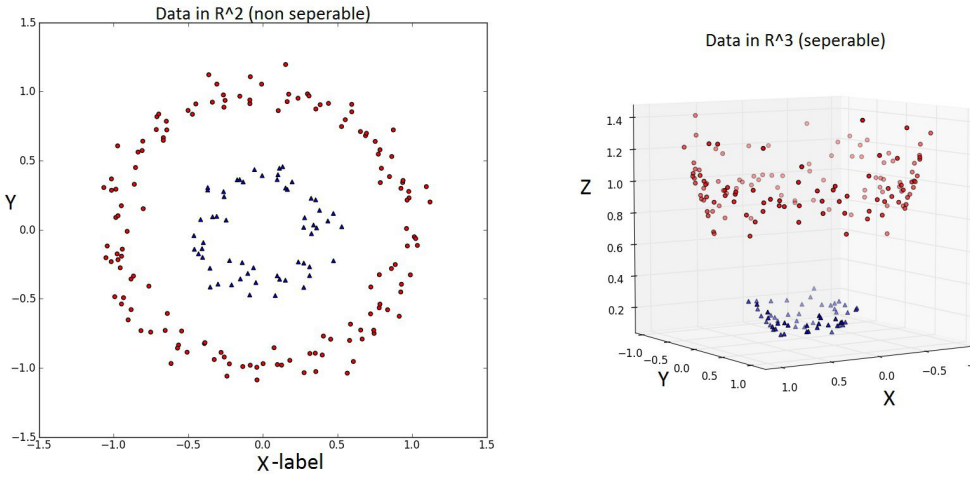

Figure 4.3: conversion of 2D data into 3D space

In the give figure 4.3 we can be observed that it is impossible to linearly separate given data in 2D. whereas after transforming it into higher dimensions makes possible to separate it by using linear methods. [21]

D) Kernel Ridge Regression(KRR): KRR differs from RR in its form of decision function. Input values are transformed into some higher dimensional feature space $X \rightarrow \Phi(X)$.

$$y_{pred} = W\Phi(X) + b \tag{4.4}$$

It is useful for problems when the features are different or larger than the number of test data points. The transformation is made computationally feasible by choice of appropriate kernels and the "kernel trick" [27]. Most common kernels are a Radial basis function (RBF) kernel, polynomial Kernel (Poly) and linear Kernel as shown in Table 4.1 [26, 27]. X and X_i are two points from the dataset. In the dual space, the

Table 4.1: some frequently used kernels are shown

RBF	$(exp(-\gamma(X - X_i)^2))$
Poly	$(XX_i + b)^d$
Linear	(XX_i)

Mathematical Learning Model is given by,

$$y_{pred} = -\frac{\alpha}{\lambda}K(X, X_i) \tag{4.5}$$
$$K(X, X_i) = \Phi(X) \cdot \Phi(X_i)$$

where K is the considered kernel and α is an array of Lagrange multipliers. KRR uses a similar loss function as ridge regression and kernel trick which has its origins in SVM. That way, it can be seen as an intermediate between Ridge regression and SVM. Kernel ridge regression requires less time to train compared to SVM [29].

By transforming space into the higher dimension, it is possible to calculate the linear relation between input(X) and output(y).Transformation of feature space can be done by selecting RBF, Poly or Linear kernel. Each kernel gives different transformation result as output because the kernel function of each kernel is different. Depending on data(whether it is in a linear, exponential or polynomial relationship) the kernel provide results. If data is in an exponential relationship and we are using the linear kernel in KRR then the algorithm will perform poorly.

E) Support Vector Machines Regression (SVM Regression):

In a nutshell, SVM regression differs from KRR in its loss function which allows it to select a subset of input data set (called support vectors) which are used in the learning model. Generally, it takes longer to train the SVM model compared to KRR but results in a compact decision function. A support vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data

point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier [33].

The type of that equation will depend on what type of kernel is used during training of algorithm using training data. Different kernels select a different number of support vectors and different number of coefficients as well.(number of coefficients are equal to number of support vectors) Coefficients will be multiplied with kernel functions in the equation of prediction. Support vectors (which is selected during training by the used kernel) will be used in decision function as data points. The test data(X) will be provided to this equation as a second input to predict the value of another quantity(y). Suppose we are given training data $(x_1, y_1), ..., (x_l, y_l)$.

The loss function of SVM Regression is,

$$min[1/2(W)^2 + [C\sum(\xi^* + \xi)]] \tag{4.6}$$

subject to:

$$[y_{real} - (W, \Phi(X)) - b] \leq \epsilon + \xi \tag{4.7}$$

$$[(W, \Phi(X)) + b - y_{real}] \leq \epsilon + \xi^* \tag{4.8}$$

$$\xi, \xi^* \geq 0 \tag{4.9}$$

Decision function/Mathematical learning model of SVM regression in dual space is:

$$\sum(\alpha - \alpha^*)K(X, X_i) + b \tag{4.10}$$

where ξ and ξ^* are slack variables to cope with constraints of the optimization. The constant $C > 0$ controls the regularization in SVR while α and α^* are Lagrange multipliers [29]. A lower C means higher regularization. A lower C thus prevents the algorithm from overfitting. Having a lower C means the optimization doesn't concerned with the loss too much.

4.5 Optimization Techniques

Gradient descent is a first-order iterative optimization algorithm for finding the minimum of a function. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient (or of the approximate gradient) of the function at the current point. If instead one takes steps proportional to the positive of the gradient, one approaches a local maximum of that function; the procedure is then known as gradient ascent. Gradient descent is also known as

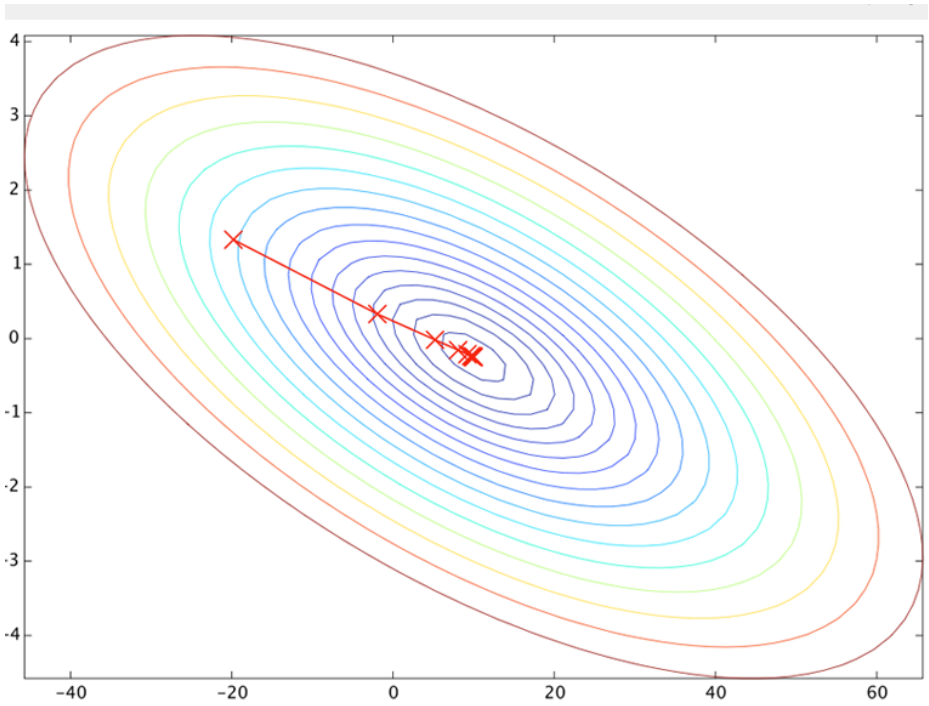

Figure 4.4: Gradient descent application in finding global minimum

steepest descent. However, gradient descent should not be confused with the method of steepest descent for approximating integrals.

The training data is used to optimize the Learning model function and obtain min. value of error function. sklearn library with python interface was used to define and train the ML regression algorithms. For optimization of parameters, we used grid-search method by using GridSearchCV library from sklearn [29]. GridSearchCV does an exhaustive search over defined parameters with a scoring method of 'mean square error'. The parameters of the estimator used to apply these methods are optimized by cross-validated grid-search over a parameter grid. Training was performed in a batch learning mode in batches of 4 and 200 epochs were performed. One complete training took less than a minute on a standard PC. Optimization of parameters are done using GridSearchCV library [29]. Given figure 4.4 shows the path chosen by Gradient descent method to find optimum parameters of machine learning algorithm.

In the figure 4.5 we want to find the minimum value of std deviation and mean of error percentage while predicting the output by using SVM Regression using radial basic kernel. In 1992, Boser, Guyan, and Vapnik proposed a way to model more complicated relationships by replacing each dot product with a nonlinear kernel function (such as a Gaussian radial basis function or Polynomial kernel). Gamma is the free parameter of the Gaussian radial basis function. A small gamma means a Gaussian

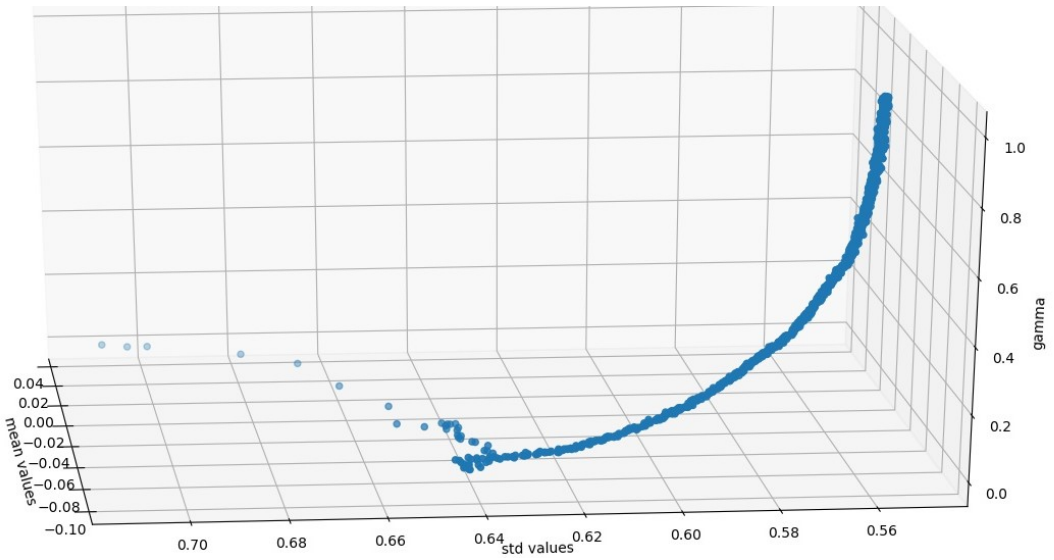

Figure 4.5: Optimization curve shows exhaustive search done for optimization of standard deviation and mean while varying all possible gamma

with a large variance so the influence of support vector is more, a small gamma implies the class of this support vector will have an influence on deciding the class of the test data vector even if the distance between them is large. If gamma is large, then the variance is small implying the support vector does not have wide-spread influence. Technically speaking, large gamma leads to high bias and low variance models, and vice-versa.

Chapter 5

Training and Testing of Machine Learning Regression Algorithms

5.1 Training

The process of training an ML model involves providing an ML algorithm (that is, the learning algorithm) with training data to learn from. The term ML model refers to the model artifact that is created by the training process. The training data must contain the correct answer, which is known as a target or target attribute. The learning algorithm finds patterns in the training data that map the input data attributes to the target (the answer that you want to predict), and it outputs an ML model that captures these patterns. One can use the ML model to get predictions on new data for which one do not know the target. For example, let's say that we want to train an ML model to predict if an email is a spam or not a spam. We would provide ML Algorithm with training data that contains emails for which we know the target (that is, a label that tells whether an email is a spam or not a spam). ML would train an ML model by using this data, resulting in a model that attempts to predict whether new email will be a spam or not a spam.

To train an ML model, you need to specify the following:

1. Input training data-source(training data)
2. Name of the data attribute that contains the target to be predicted
3. Required data transformation instructions

Training parameters to control the learning algorithm. The training of ML regression algorithms is performed with three distinct parameter sources, measured profile (100 points in each profile), particle number N_p (1 point) and bunch length σ_l (1 point) as inputs forming an array of 102 points for each training sample while the output is actual width denoted by $\sigma_{x,a}$ to differentiate from predicted width $\sigma_{x,p}$. $\sigma_{y,a}$

is not used for the training process since in any experimental usage, it will not be available as an input to the trained network for prediction of $\sigma_{x,p}$. 375 training samples were generated with a parameter scan in N_p, $\sigma_{x,a}$ and $\sigma_{y,a}$ (5 parameters each) and σ_l (3 parameters). Table 3.1 shows the parameter range over which training data was generated. Validation data was generated at a spacing $1\%, 25\%$ and 50% off the training data sites in each parameter space forming 372 validation samples. In addition to that 0.5% Gaussian white noise (relative to the maximum value in each parameter space) was added to the profiles to depict ADC/camera noise on the measured profile as well as measurement uncertainty on N_p and σ_l. More details can be found in [31] The training data is used to optimize the loss function and obtain the parameters of regression algorithms. sklearn library with python interface was used to define and train the ML regression algorithms discussed above. For optimization of parameters, we used gridsearch method by using GridSearchCV library from sklearn. GridSearchCV does an exhaustive search over defined parameters with a scoring method of 'mean square error' [29].

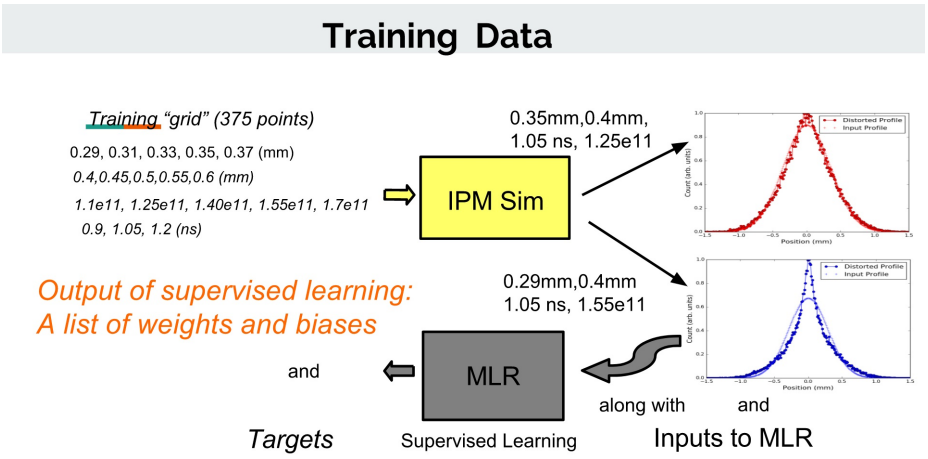

Figure 5.1: The way training data is provided to Machine learning Algorithm

5.2 Results on Test Data

For every ML Algorithm, the error% between reconstructed width and true width of the profile is calculated for the whole validation data and plotted as a histogram. All results are generated by adding Gaussian white noise=0.5% with respect to source input parameter.

5.3 Linear Regression

A) Linear Regression: We trained Linear Regression using training data and tested with validation data. Figure 5.2 shows the Histogram of error%. Figure 5.4 shows the reconstructed beam width for the validation data by the trained Linear Regression Algorithm plotted against actual (initial) beam width. Figure 5.3 Gaussian white noise was added to each channel on the Linear regression input for both training and validation data in the range of σ_{noise} = 0 % to 10 % relative to the maximum value to that parameter. For each set of noisy training data, the Linear regression training was performed 10 times and the bias and variance of prediction error were plotted against the added noise as shown in Figure 5.3. Initially, variance drops abruptly, after that A linear trend is seen in the increase of prediction error variance and bias hovers around zero.

Figure 5.2 shows the histogram of error% in prediction by linear regression.

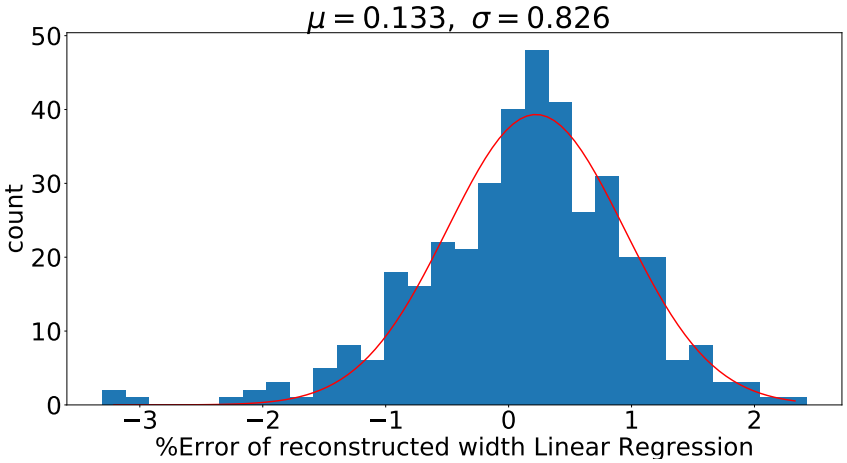

Figure 5.2: Histogram showing the percentage prediction error of approximated function by Linear Regression Algorithm.

where $\mu = 0.144$ and $\sigma = 0.860$

Figure 5.3 shows bias and variance % in prediction by linear regression.

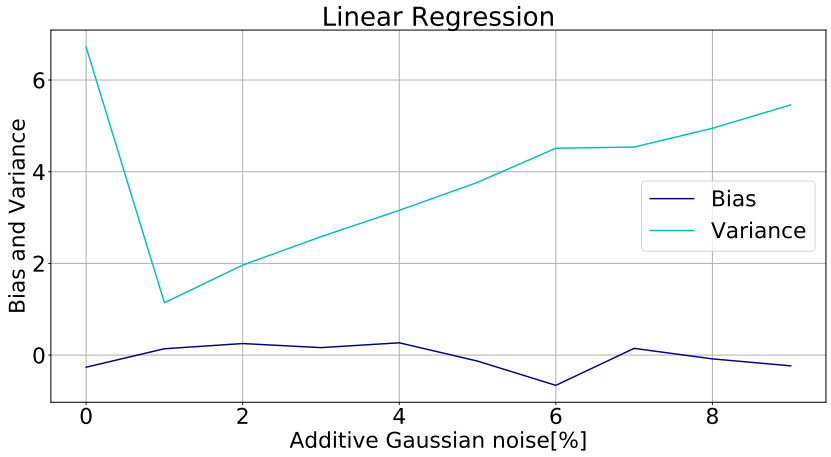

Figure 5.3: Evolution of bias and std. deviation of predictions with respect to noise in training and validation data using Linear Regression Algorithm.

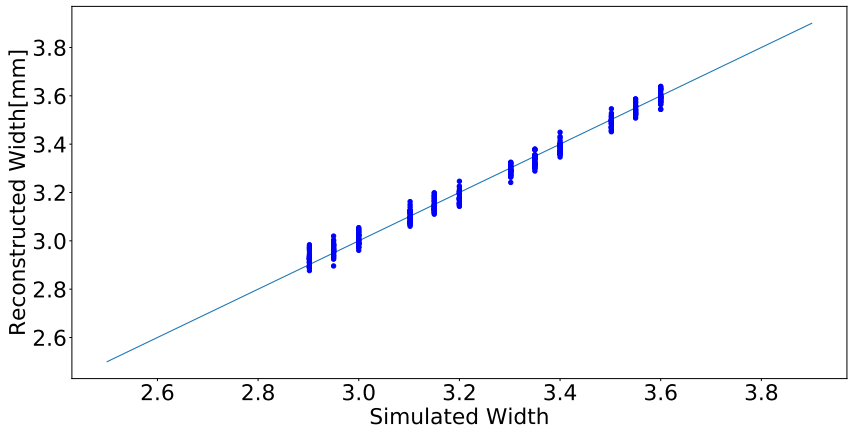

Figure 5.4: shows the reconstructed beam width for the validation data by the trained Linear Regression Algorithm plotted against actual (initial) beam width

5.4 Ridge Regression

B) Ridge Regression: In Ridge Regression after optimization using Gridsearch method we obtained $\lambda = 0.0189$. We trained Ridge Regression using training data and tested with validation data.Figure 5.6 shows the Histogram of error% which is error% between reconstructed width and true width of the profile.

Figure 5.7 shows the reconstructed beam width for the validation data by the trained Ridge Regression Algorithm plotted against actual (initial) beam width.

Gaussian white noise was added to each channel on the Linear regression input for both training and validation data in the range of $\sigma_{noise} = 0$ % to 10 % relative to the maximum value to that parameter. For each set of noisy training data, the Linear regression training was performed 10 times and the bias and variance of prediction error were plotted against the added noise as shown in Figure 5.8.A linear trend is seen in the increase of prediction error variance and bias hovers around zero.

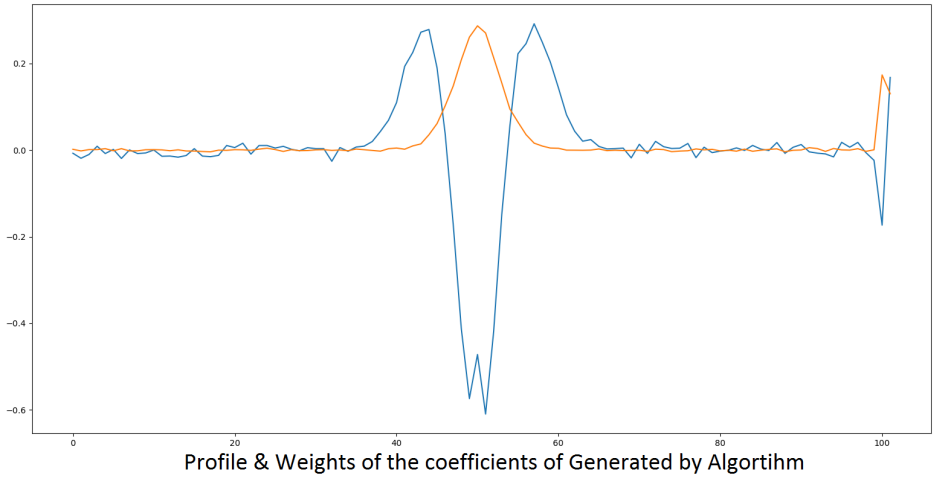

Figure 5.5: weight of regression algorithm plotted with the profile

In given figure 5.5 the weights of generated equation(Blue) is plotted with the original profile(Orange). This helps us determine which points or areas of curve plays an important role while predicting/reconstructing the original profile. As well as which points are extracted by the algorithm as effective features.

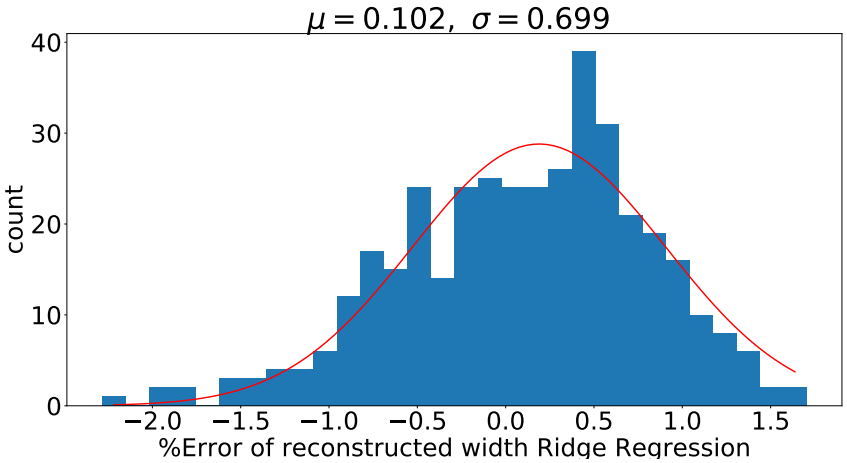

Figure 5.6: Histogram showing the percentage prediction error of approximated function by Ridge Regression Algorithm

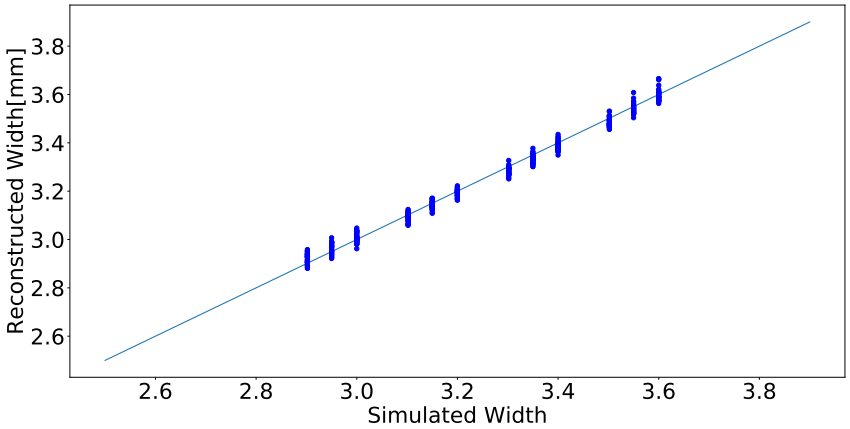

Figure 5.7: Prediction of actual profile width from distorted measured profile using Ridge Regression Algorithm

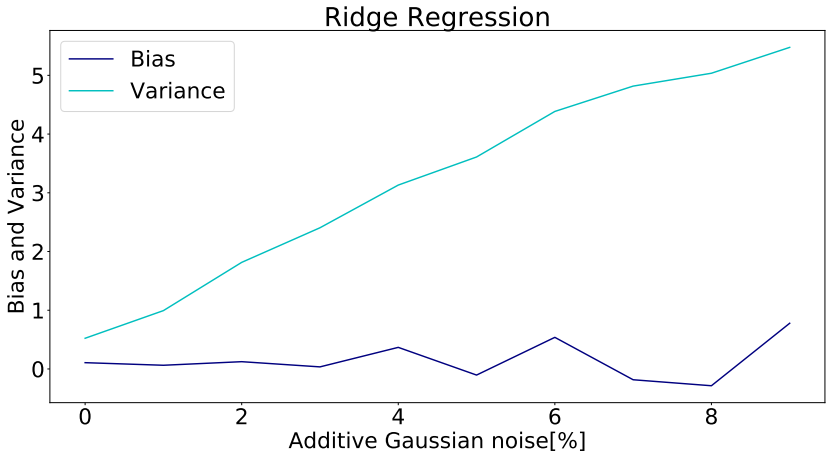

Figure 5.8: Evolution of bias and std. deviation of predictions with respect to noise in training and validation data using Ridge Regression Algorithm

5.5 Kernel Ridge Regression

Kernel Ridge Regression: Different results are obtained when we use different Kernel for training and testing of data.So we optimized parameters for each kernel of KRR using Gridsearch method. For 'RBF' kernel we optimized parameters as $\lambda = 0.00106$, $\gamma = 0.1$.Histogram Figure 5.12 shows error% between Predicted width and true width of the profile.

Figure 5.13 shows the reconstructed beam width for the validation data by the trained Kernel Ridge Regression Algorithm(RBF) plotted against actual (initial)

beam width.

Gaussian white noise was added to each channel on the Kernel ridge regression input for both training and validation data in the range of σ_{noise} = 0 % to 10 % relative to the maximum value to that parameter. For each set of noisy training data, the Kernel ridge regression(RBF kernel) training was performed 10 times and the bias and variance of prediction error were plotted against the added noise as shown in Figure 5.14.A linear trend is seen in the increase of prediction error variance and bias hovers around zero.

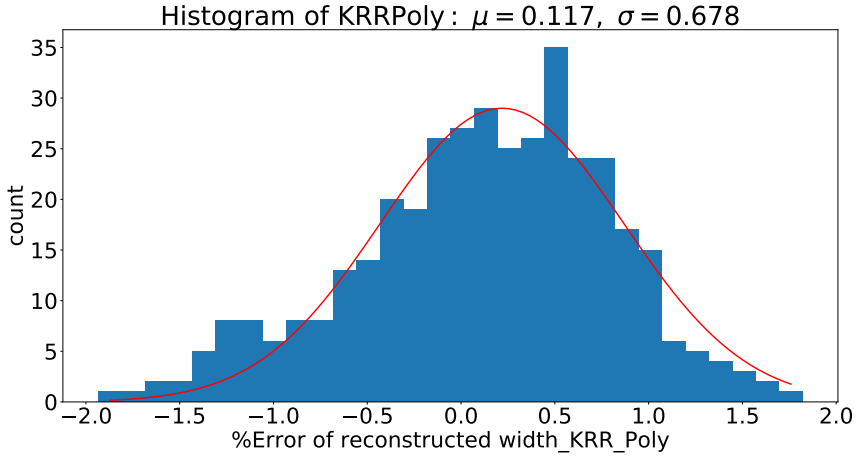

Figure 5.9: Histogram showing the percentage prediction error of approximated function by Kernel Ridge Regression Algorithm using Poly Kernel

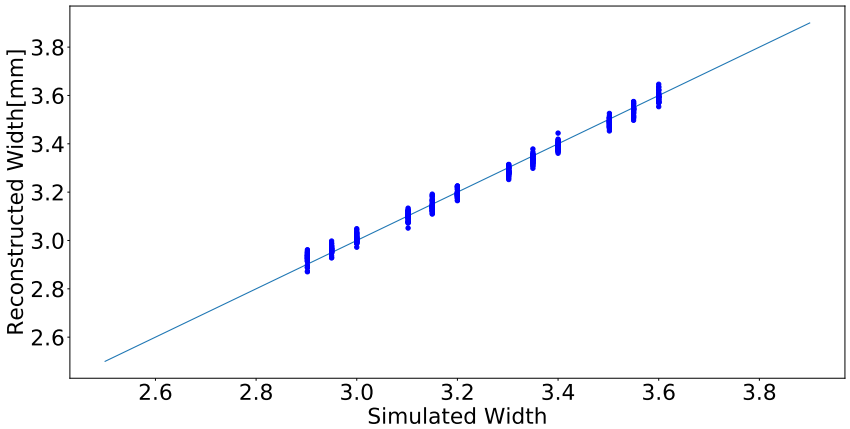

Figure 5.10: Prediction of actual profile width from distorted measured profile using Kernel Ridge Regression Algorithm using Poly Kernel

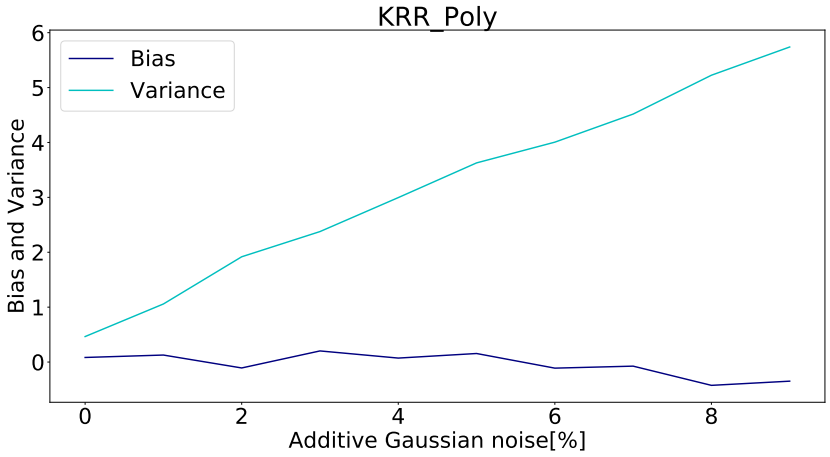

Figure 5.11: Evolution of bias and std. deviation of predictions with respect to noise in training and validation data using Kernel Ridge Regression Algorithm using Poly Kernel

For KRR using Polynomial kernel, we optimized parameters of the kernel using Gridsearch method. For 'Poly' kernel we optimized parameters as $\lambda = 0.0005$, $\gamma = 0.01$.The histogram in Figure 5.9 shows error% between Predicted width and true width of the profile.

Figure 5.13 shows the reconstructed beam width for the validation data by the trained Kernel Ridge Regression Algorithm(RBF) plotted against actual (initial) beam width.

Gaussian white noise was added to each channel on the Kernel ridge regression input for both training and validation data in the range of $\sigma_{noise} = 0$ % to 10 % relative to the maximum value to that parameter. For each set of noisy training data, the Kernel ridge regression(RBF kernel) training was performed 10 times and the bias and variance of prediction error were plotted against the added noise as shown in Figure 5.14.A linear trend is seen in the increase of prediction error variance and bias hovers around zero.

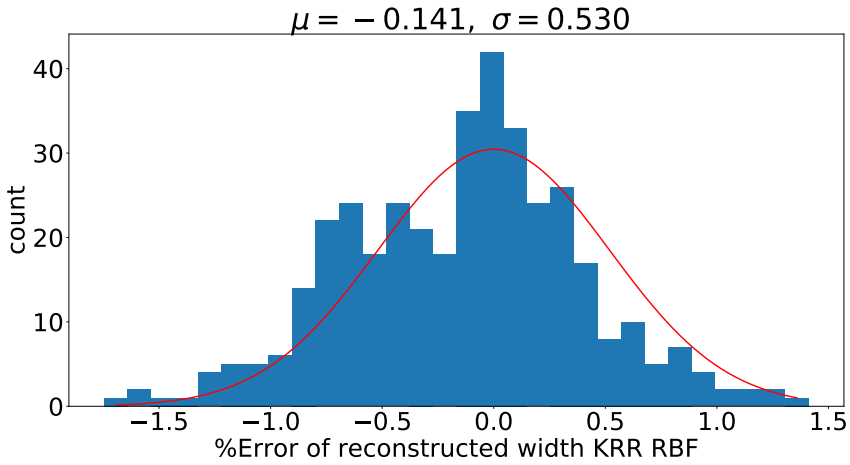

Figure 5.12: Histogram showing the percentage prediction error of approximated function by Kernel Ridge Regression Algorithm using RBF Kernel

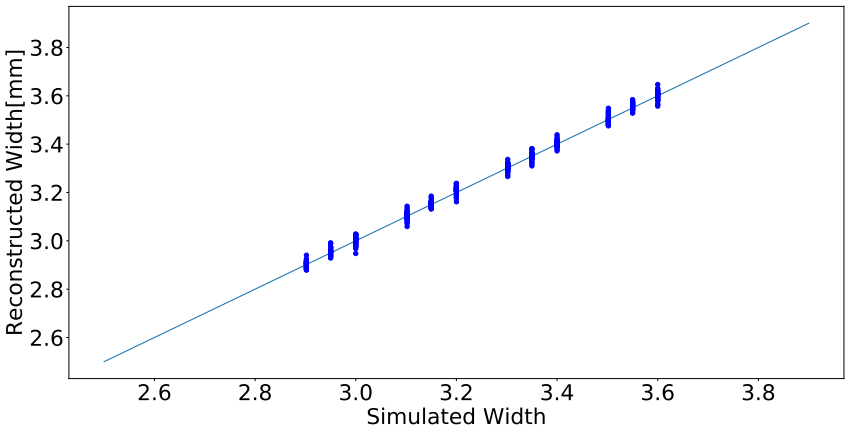

Figure 5.13: Prediction of actual profile width from distorted measured profile using Kernel Ridge Regression Algorithm using RBF Kernel

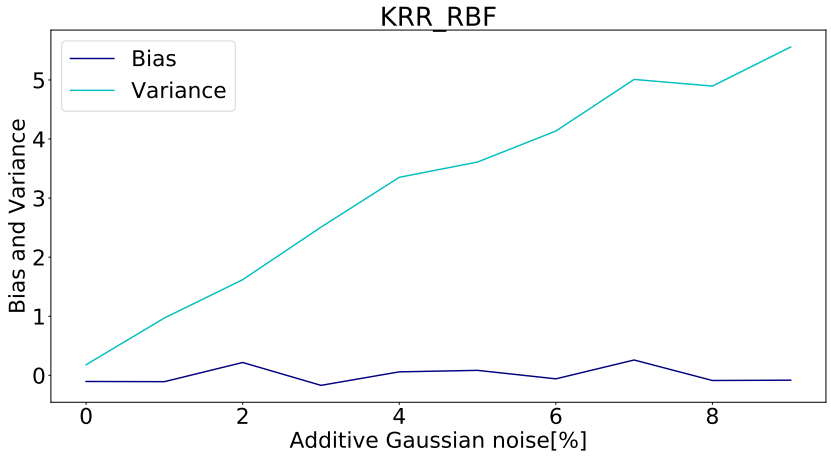

Figure 5.14: Evolution of bias and std. deviation of predictions with respect to noise in training and validation data using Kernel Ridge Regression Algorithm using RBF Kernel

5.6 Support Vector Machines

Support Vector Machines Regression: similar to KRR, we optimized SVM Regression for different kernels using Gridsearch method.For 'RBF' kernel we optimized parameters as $\epsilon = 0.024$, $\alpha = 0.1$, $C = 1000$. Histogram in Figure 5.15 shows error% between Predicted width and true width of profile.

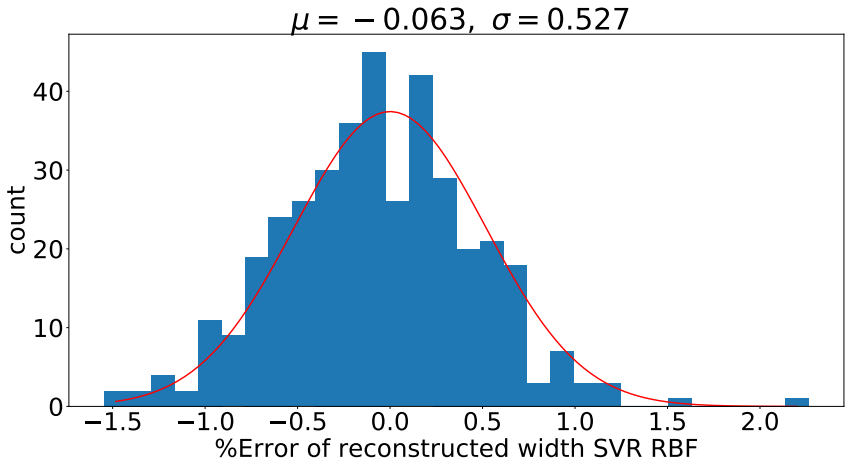

Figure 5.15: Histogram showing the percentage prediction error of approximated function by SVM Regression Algorithm using RBF Kernel

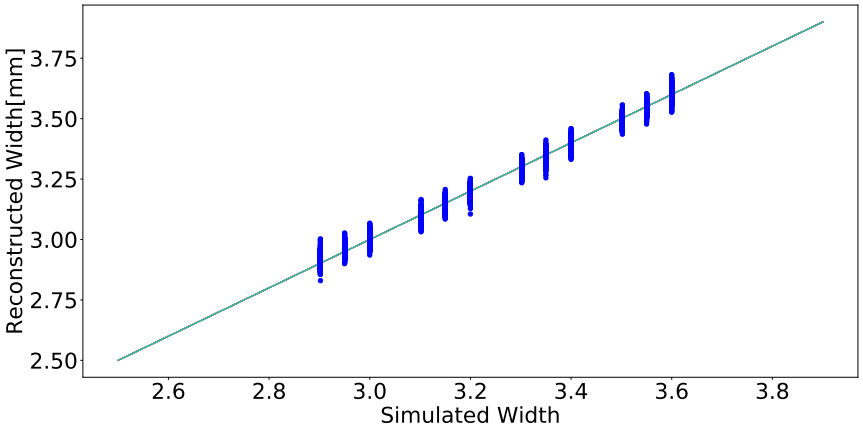

Figure 5.16: Prediction of actual profile width from distorted measured profile using SVM Regression Algorithm using RBF Kernel

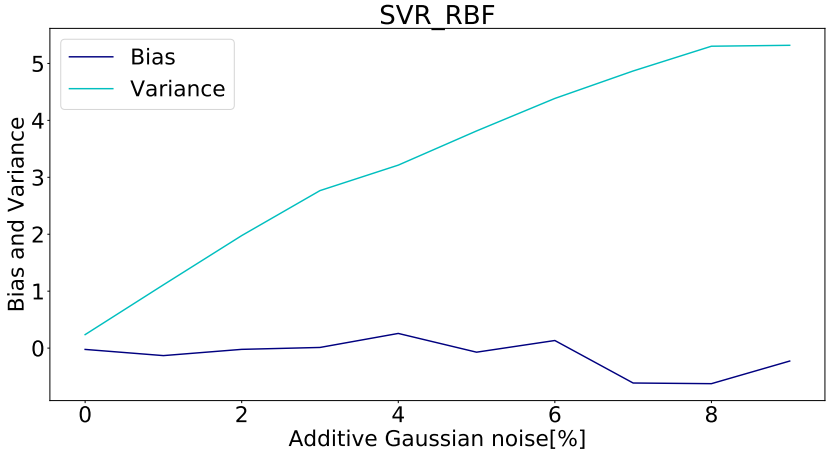

Figure 5.17: Evolution of bias and std. deviation of predictions with respect to noise in training and validation data using SVM Regression Algorithm using RBF Kernel

For 'Poly' kernel we optimized parameters as $\epsilon = 0.0120$, $\alpha = 0.1$, $C = 1000$.Histogram Figure 5.18 shows error% between Predicted width and true width of profile.

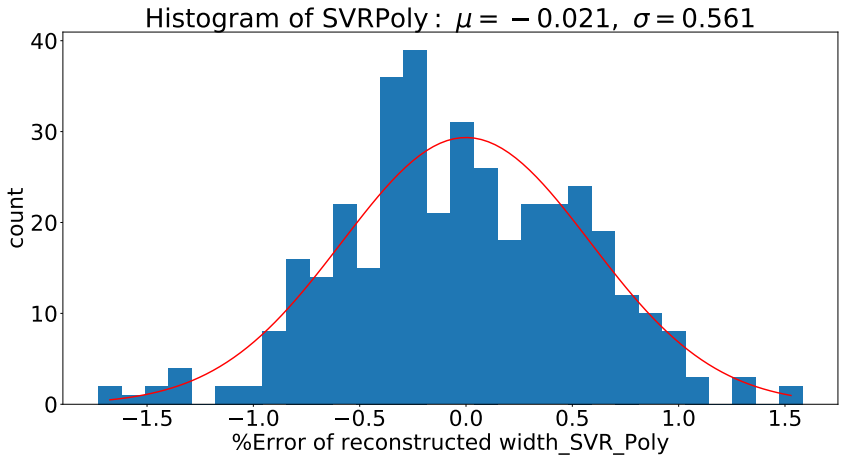

Figure 5.18: Histogram showing the percentage prediction error of approximated function by SVR Regression Algorithm using Poly Kernel

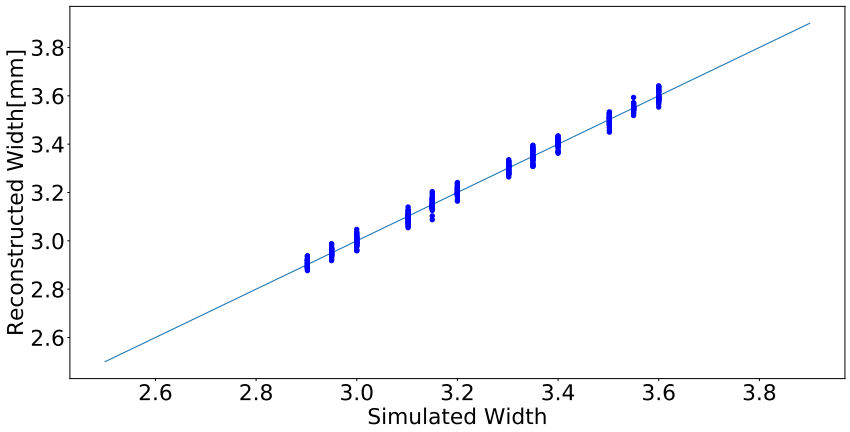

Figure 5.19: Prediction of actual profile width from distorted measured profile using SVM Regression Algorithm using Poly Kernel

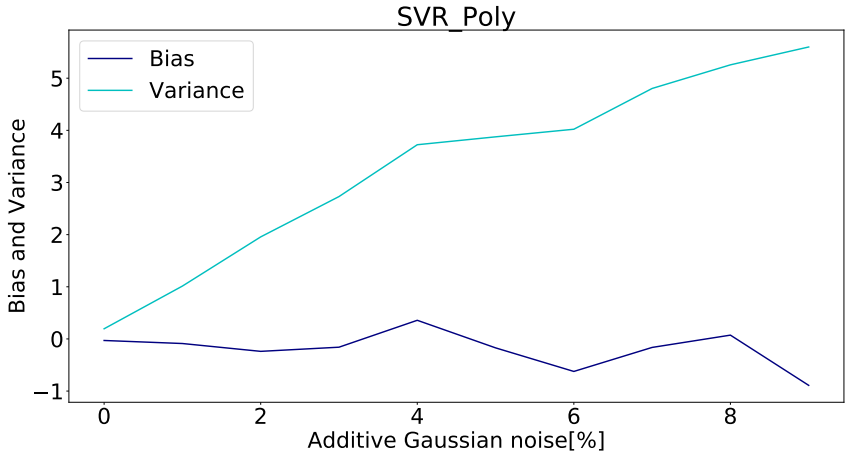

Figure 5.20: Evolution of bias and std. deviation of predictions with respect to noise in training and validation data using SVM Regression Algorithm using Poly Kernel

As we can see the deviation of error is decreased from linear Regression to SVM Regression. The value of deviation is similar to KRR-RBF(Figure 5.12) and SVR-RBF(Figure 5.15).

5.7 Combined Results

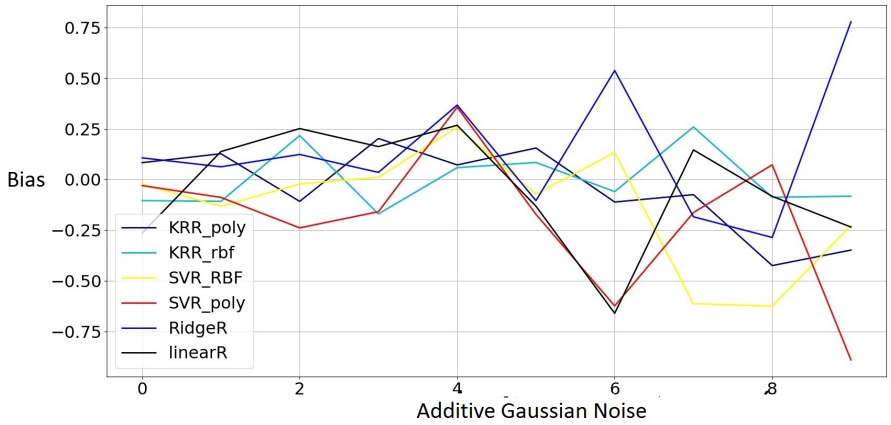

Figure 5.21: Evolution of bias of predictions with respect to noise in training and validation data

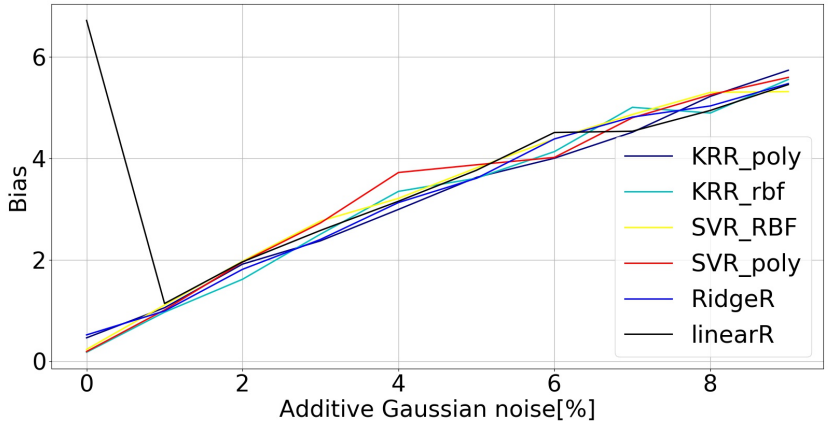

Figure 5.22: Evolution of std. deviation of predictions with respect to noise in training and validation data

Gaussian white noise was added to both training and validation data in the range of $\sigma_{noise} = 0$ % to 10 % relative to the maximum value of each source parameter. For each set of noisy training data, training of all the ML algorithms was performed and the bias and variance of prediction error over the validation data were plotted against the added % noise. Figures 5.21 and 5.22 shows plot of bias and variance against % noise respectively.

In Figure 5.22 of variance against noise we can see that almost all variances of prediction error generated by ML algorithms are in a linear relationship with % noise added to data.

The bias of the prediction error of almost all algorithms oscillates around zero even with increased noise while the variance of the prediction error is in a linear relationship with noise amplitude. As % noise in test data is increased, the bias of prediction error of SVR using the polynomial kernel, ridge regression, and linear regression oscillates rigorously around zero. Whereas KRR using the polynomial kernel, SVR using RBF kernel and KRR using RBF kernel oscillates around zero less rigorously, which means that they can perform well with noisy data.

Chapter 6

Conclusion

Measuring the transverse beam size in the Large Hadron Collider by using Ionization Profile Monitors is a difficult task for energies above injection during the energy ramp from 450 GeV to 6.5 TeV. Depending on the initial momentum distribution of the electrons, emerging from the ionization process with the highly relativistic beam, the profile distortion is affected significantly. Beam characteristics required for the accelerator design include not only rather obvious quantities like beam current or transverse position and size, but also quantities like its longitudinal size, angular dispersion, energy spread, or the number of lost particles in the accelerator that have to be known with astonishing precision. For That, The reconstruction of distorted profiles for Gaussian beams was performed using machine learning regression algorithms and reconstruction errors below 1% were obtained even with the inclusion of measurement uncertainties. Results show that compared to other machine learning regression algorithms, both Support Vector Machines Regression and Kernel Ridge Regression with RBF kernel gives better prediction results. Regression methods like linear regression and ridge regression lack behind in performance when compared with Support Vector Machines and Kernel Ridge Regression due to absence of kernels. This might be due to the fact that squared exponential kernel defines a function space that is a lot larger than that of the linear kernel or the polynomial kernel. Results from this project are going to be implemented in various particle accelerator facilities across the world.

References

[1] R. E. Thern, "Space-charge distortion in the brookhaven ionization profile monitor", PAC, 1987.

[2] Amundson et al., Calibration of the Fermilab Booster ionization profile monitor, PhysRevSTAB.6.102801 (2003)

[3] J. Egberts, IFMIF-LIPAc Beam Diagnostics: Profiling and Loss Monitoring Systems, PhD thesis, University Paris Sud, (2012)

[4] M. Sapinski et al., The first experience with LHC beam gas ionization monitor, Proceedings of IBIC 2012.

[5] M. Patecki et al., Electron tracking simulations in the presence of the beam and external fields, Proceedings of IPAC 2013, Shanghai, China.

[6] D. Vilsmeier et al., Investigation of the effect of beam space charge on electron trajectories in Ionization profile monitors, HB 2014.

[7] D. Vilsmeier et al., A modular application for IPM simulations, these proceedings.

[8] Livingston, M. S.; Blewett, J. (1969). Particle Accelerators. New York: McGraw-Hill. ISBN 1-114-44384-0.

[9] Witman, Sarah. "Ten things you might not know about particle accelerators". Symmetry Magazine. Fermi National Accelerator Laboratory. Retrieved 21 April 2014.

[10] Humphries, Stanley (1986). Principles of Charged Particle Acceleration. Wiley-Interscience. p. 4. ISBN 978-0471878780.

[11] Feder, T. (2010). "Accelerator school travels university circuit" (PDF). Physics Today. 63 (2): 20. Bibcode

[12] Nagai, Y.; Hatsukawa, Y. (2009). "Production of 99Mo for Nuclear Medicine by 100Mo(n,2n)99Mo". Journal of the Physical Society of Japan. 78 (3): 033201.

[13] M. Sapinski et al., Ionization profile monitor simulations - status and future plans, Proceedings of IBIC 2016.

[14] P. Forck, Lecture notes in beam instrumentation, JUAS, 2017.

[15] K. P. Murphy, Machine Learning: A Probabilistic Perspective, The MIT Press, 2012.

[16] A. Voitkiv et al., Hydrogen and helium ionization by relativistic projectiles in collisions with small momentum transfer, J. Phys. B: At. Mol. Opt. Phys, vol. 32, 1999.

[17] K. Hornik "Approximation Capabilities of Multilayer Feedforward Networks", Neural Networks, 4(2), 251257, (1991).

[18] Chollet, François et al., Keras, 2015

[19] M. Abadi et al., TensorFlow: Large-scale machine learning on heterogeneous systems, 2015.

[20] Supposedly paraphrased from: Samuel, Arthur (1959). "Some Studies in Machine Learning Using the Game of Checkers". IBM Journal of Research and Development. 3 (3). doi:10.1147/rd.33.0210.

[21] Python Machine Learning by Sebastian Raschka.

[22] K. P. Murphy, Machine Learning: A Probabilistic Perspective, The MIT Press, (2012).

[23] Bishop, Christopher (2006). Pattern recognition and machine learning. Berlin: Springer. ISBN 0-387-31073-8.

[24] Ryan Tibshirani, Modern regression 1: Ridge regression Data Mining: 36-462/36-662 (2013). retrieved from: `http://www.stat.cmu.edu/ ryantibs/datamining/lectures/16-modr1.pdf`

[25] Stefan Feuerriegel, Regularization Methods Business Analytics Practice Winter Term (2015/16). retrieved from: `https://www.is.uni-freiburg.de/resources/business-analytics/XX_Regularization.pdf`

[26] P.Paisitkriangkrai, Linear Regression and Support Vector Regression (2012). retrieved from:
http://cs.adelaide.edu.au/ chhshen/teaching/ML_SVR.pdf

[27] Justin Domke, Statistical Machine Learning Kernel Methods and SVMs, UMass (2011). retrieved from:
https://people.cs.umass.edu/ domke/courses/sml2011/07kernels.pdf

[28] K. P. Murphy, Machine Learning: A Probabilistic Perspective, The MIT Press, (2012).

[29] Scikit-Learn Documentation 0.19.0(2017). retrieved from:
http://scikit-learn.org/stable/
modules/svm.html.

[30] David A. Freedman, Statistical Models: Cambridge University Press.p.26. ISBN: 9780521112437, (2009).

[31] R. Singh et al., Proceedings of IBIC (2017).

[32] Scikit-Learn Kernel Ridge Regression(2017). retrieved from :http://scikit-learn.org/stable/modules/kernel_ridge.html

[33] Wikipedia:Support Vector Machines Definition, retrieved from :
https://en.wikipedia.org/wiki/Suppor
t_vector_machine

[34] JCGM 200:2008 International Vocabulary of Metrology Basic And General Concepts And Associated Terms (VIM)(2008). retrieved from:
http://www.bipm.org/utils/common/
documents/jcgm/JCGM_200_2008.pdf

[35] Mehryar Mohri,Introduction to Machine Learning, Courant Institute and Google Research,retrieved from:
http://www.cs.nyu.edu/ mohri/mlu/
mlu_lecture_1.pdf