Prediction of Decay Modes of Higgs Boson using Classification Algorithms

Mohit Baliyan[1,a *], Akshat Bandooni[2,b], Sharad[3,c] and Akshansh[4,d]

[1]IMSEC, AKTU University, Ghaziabad, India

[2]IMSEC, AKTU University, Ghaziabad, India

[3]IMSEC, AKTU University, Ghaziabad, India

[4]IMSEC, AKTU University, Ghaziabad, India

[a]mohitbaliyan98@gmail.com, [b]Akshatbandooni333@gmail.com, [c]skumarchaudhary12345@gmail.com and [d]akshanshsingh97@gmail.com

Keywords: Higgs Boson, Classifiers, Decay Modes, CERN, GPU

Abstract. Computational algorithms have been applied to several problems in high energy physics from explosion of applications in particles to event identification and reconstruction. The motive of this study is to propose a model which predicts the decay events of Higgs Boson as "tau-tau decay of Higgs Boson" or "background noise" after applying some features extraction on the CERN dataset. Therefore, eight algorithms namely K Nearest Neighbors, Naïve Bayes, Support Vector Machine, Artificial Neural Network, Logistic Regression, Decision Tree, Random Forest and Gradient Boosting are used in this experiment. The performances of all the algorithms are evaluated on measures like Accuracy and Computational Time. Results obtained show Decision Tree outperforms the best on the measures like Accuracy and Computational Time.

Introduction

On 4 July 2012 The European Organization for Nuclear Research (CERN) observed a heavy particle around 126 MEV which assumed to be consistent with the long-sought Higgs Boson and Later in March 2013 the detection of a new particle was confirmed and New results indicate that new particle is a Higgs boson [1] which was verified by CERN. This ATLAS experiment takes place at the CERN's large Hadron Collider. LHC is the most powerful particle accelerator. To boost the energy of the particles, it has a 27 km ring of superconducting magnets. Before they are made to collide, two high energy particle beams travel inside the accelerator with the speed close to the light. Strong magnetic fields produced by the superconducting electromagnets are used to guide the accelerator ring. Beams inside the accelerator ring collide at the position of ATLAS, CMS, ALICE and LHCB particle detectors. ATLAS is a general purpose detector at the LHC. Beams of particles from The Large Hadron Collider collide at its center Deflection of new particles from the collision point fly in all directions [2]. Paths, momentum and the energy of the particles is recorded by the ring detector subsystems which are arranged in layers. ATLAS creates a huge amount of flow data and uses an advanced trigger system to record the worth events. The existence of Higgs Boson is also confirmed on the basis of collisions at the ATLAS [3]. The Higgs is one of the 17 particles in the Standard Model which describes the all known basic particles. The Higgs particle is a Boson which is responsible for all physical forces and giving particles their mass. It is hard to detect Higgs Boson, as it is very massive compared to other particles and does not last very long and it obeys the law of conservation of energy.

Due to the short lifetime of Higgs Boson, it disintegrates into other particles. Higgs Boson decays in different types of elementary particles that carry forces. A decay of Higgs Boson into specific particles is called channels. Three distinct decay channels were observed in Higgs Boson which were all boson pairs. The next decay of Higgs Boson has been seen into the fermion pairs, namely

as tau-leptons or b-quarks. One of the ATLAS experiments showed that Higgs Boson decays into two tau particles. Tau particles belong to a group of subatomic particles called fermions, which are responsible for making up the matter. Tau is a Fermion. Fermions behave like a very massive electron and can be an electron or any composite particle such as the proton. They obey the Pauli exclusion principle and have half integer spin. Fermions acquire mass in the same manner, how gauge bosons acquire the mass that's why Higgs Boson could decay to either bosons or fermions. But the strength of this signal is small, that's why it is buried in the background noise. The simulated data is created by ATLAS after detecting the features which are responsible to characterize the events. So, each event is classified in two categories, one is "tau-tau decay of Higgs Boson" and other one is background noise.
This Paper, firstly performed data analysis on this CERN dataset and on the basis of this analysis it preprocessed the data. Then applied several classifiers on this data to classify events as decay of Higgs Boson into tau particles or as background noise and evaluate the performance of each classifier. At the end performed time analysis to compare the GPU based computation for each classifier over training and test data.

Literature Review

Machine learning plays a significant role in processing the large data at particle colliders. Energy physics takes advantage of Machine learning at two levels, the online filtering of streaming detector measurements and as well as offline analysis of recorded data. Particle colliders sometimes do not produce particles of interest. Supervised machine learning has been used to separate the events producing the particles of interest (signal) from the events producing other particles (background). According to Whiteson stochastic optimization techniques result in substantially better analysis [4]. Artificial neural networks have been used to detect the decay of Higgs Boson to tau leptons, as deep neural network architectures have the ability to automatically discover high level features from the data. According to Bruce Denby machine learning increases the statistical power more than the common high-level features handcrafted by the physicists [5]. Approaches of machine learning are often used for searching the rare particles produced during the collision. Standard machine learning approaches like shallow machine learning models have a capacity of learning complex linear functions of the inputs. On the other hand, as explained by Baldi, Sadowski, & Whiteson to learn complex non-linear functions of inputs, artificial neural networks have the ability to discriminate the signals of decay better from the background noise [6]. Stacking machine learning algorithms performed worse than deep neural networks but with a less computation cost. However, stacked classifiers in a multivariate statistical analysis (MVA) showed a significant enhancement. Alves suggests that MVA tools for ensemble of simpler and faster machine learning algorithms performs better than building a complex state of art algorithm for cut and count [7].

Methodology

Dataset: The dataset is permanently released at CERN Open Data Portal [8]. The ground truth data has 818238 events in 195.5 MB. The dataset Higgs Boson Training Data which is used in our experiment is downloaded from Kaggle [9]. It is composed of two files, training file and test file. Overview of the structure of the dataset is described in Table 1. As the label column is not present in the Test data file as a consequence Training data file is used in this experiment.

Table 1: Dataset Description

Observations	Training file	Test file
Events	250000	550000
ID Column	Yes	Yes
Feature Column	30	30
Weight	Yes	Yes
Label Column	Yes	Yes

Data Preprocessing: Value -999.000 represents the non-computed value in the dataset. Number of non-computed values in the different feature columns is described in Table 2.

Table 2: Number of Non-Computed Values

Column Name	Number of non-computed values
DER_mass_MMC	38114
DER_deltaeta_jet_jet	177457
DER_mass_jet_jet	177457
DER_prodeta_jet_jet	177457
DER_lep_eta_centrality	177457
PRI_jet_leading_pt	99913
PRI_jet_leading_eta	99913
PRI_jet_leading_phi	99913
PRI_jet_subleading_pt	177457
PRI_jet_subleading_eta	177457
PRI_jet_subleading_phi	177457

Most columns present in Table 2 contain more than the half number of entries in non-computed form. To deal with non-computed values, first calculate the correlation and the dependencies of each column to each other. Fig. 1. shows the correlation between these columns. As shown in Fig. 1. Variable DER_mass_MMC does not highly correlate with any other variable and the number of non-computed values in this variable is less than half of its total entries. We didn't drop the column of this variable at the cost of information loss so we replaced its non-computed values by the mean of this column.

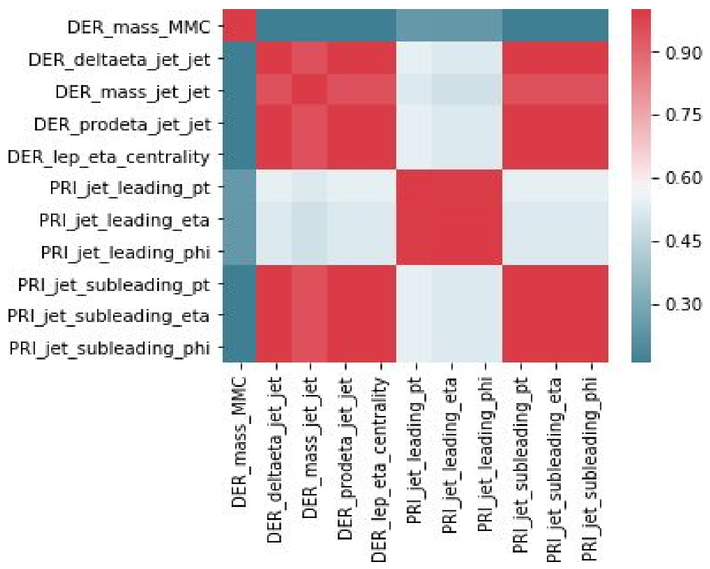

Figure 1: Correlation between these columns

PRI_jet_leading_phi and PRI_jet_leading_eta variables are highly correlated with PRI_jet_leading_pt and have non-computed entries in more than half no of rows so we dropped these two columns to avoid uncertainty during training and keep PRI_jet_leading_pt to avoid information loss. Variables DER_lep_eta_centrality, DER_mass_jet_jet, DER_prodeta_jet_jet, PRI_jet_subleading_eta, PRI_jet_subleading_phi and DER_deltaeta_jet_jet is highly correlated to PRI_jet_subleading_pt and contain more than half entries in non-computed form so we dropped them and keep PRI_jet_subleading_pt to keep the features information of dropped columns, where min is the minimum value in that feature vector and max is the maximum value in that feature vector.

The variables prefixed with PRI (for Primitives) are raw quantities about the bunch collision as measured by the detector, essentially the momenta of particles. Variables prefixed with DER (for Derived) are quantities computed from the primitive features. Fig 2 showed the total number of counts of “tau-tau decay of Higgs Boson” and “Background noise” in the target variable for each event where “s” represents the signal of decay and “b” represents the background noise.

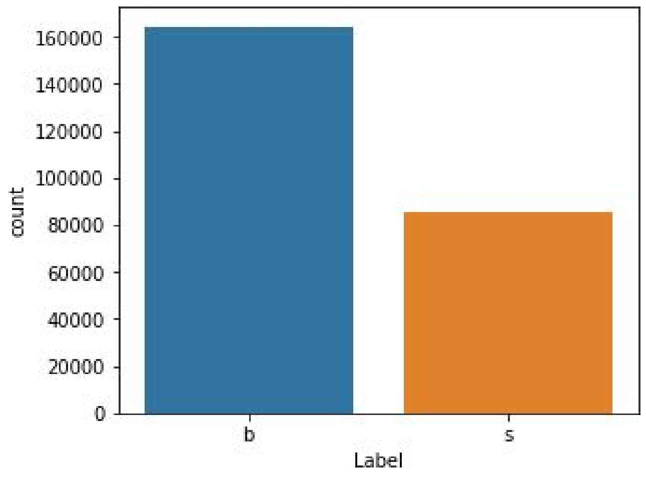

Figure 2: Total Count

There is an unbalanced distribution in event classification as we have a greater number of events which are classified as "background noise" in comparison to "tau-tau decay of Higgs Boson". Label column is the target variable which is categorical. It contains entries in two categories ’s’ for ’tau-tau decay of Higgs Boson’ and ’b’ for ’background noise’. So, we encoded the label column in the form of 0 and 1 to make it non-categorical.
Classifiers like logistic regression and neural networks work well on normalized inputs. In case of logistic regression and neural networks, it ensures the stability in convergence of weights and bias during optimization of cost function. For k-nearest neighbor, it has two aspects, one is to use the Euclidean distance between two entries of different range make our classifier uninformative and according to second aspects if we didn’t normalize the inputs for k-nearest neighbor, all nearest neighbors are aligned near the same axis which has smaller range and leads to incorrect classification. But in our experiment K-nearest neighbor works better on second aspects, so we feed the normalized data into k-nearest neighbor classifier. We used a Min-Max scaler to normalize our data whose value for any entry is given by equation 1.

$$x^J = (x - \min)/(\max - \min) \quad (1)$$

After data preprocessing, we have 24 features left for characterizing the events as "tau-tau decay of Higgs Boson" or "background noise" in our dataset. Feature vector removed after preprocessing refer with Table 3.

Table 3: Characteristics of Feature columns

Column Name	Characteristics
DER_mass_MMC	The estimated mass mH of the Higgs boson candidate, obtained through a probabilistic phase space integration.
DER_deltaeta_jet_jet	The absolute value of the pseudorapidity separation between the two jets (undefined if PRI$_j$*et*$_n$*um*1).
DER_mass_jet_jet	The invariant mass of the two jets (undefined if PRI$_j$*et*$_n$*um*1).
DER_prodeta_jet_jet	The product of the pseudorapidities of the two jets (undefined if PRI$_j$*et*$_n$*um*1).
DER_lep_eta_centrality	The centrality of the pseudorapidity of the lepton w.r.t. the two jets (undefined if PRI$_j$*et*$_n$*um*1).
PRI_jet_leading_pt	The transverse momentum of the leading jet, that is the jet with largest transverse momentum (undefined if PRI$_j$*et*$_n$*um* = 0).
PRI_jet_leading_eta	The pseudorapidity of the leading jet (undefined if PRI jet num = 0).
PRI_jet_leading_phi	The azimuth angle of the leading jet (undefined if PRI jet num = 0).
PRI_jet_subleading_pt	The transverse momentum of the leading jet, that is, the jet with second largest transverse momentum (undefined if PRI$_j$*et*$_n$*um*1).
PRI_jet_subleading_eta	The pseudorapidity of the subleading jet (undefined if PRI$_j$*et*$_n$*um*1).
PRI_jet_subleading_phi	The azimuth angle of the subleading jet (undefined if PRI$_j$*et*$_n$*um*1).

Method Used

Classifiers are the models which learn from the data to map the input pattern to a specific class. We use various classifiers to evaluate the performance of each classifier on our data which are given below:

K-Nearest Neighbor Classification (KNN) is one of the simplest and fundamental classifiers and performs exceptionally well when there is no prior or little knowledge about the distribution of data. K-nearest-neighbor classification was developed from the need to perform discriminant analysis when reliable parametric estimates of probability densities are unknown or difficult to determine [10].

K-nearest neighbors make the prediction by using the training data directly, so there is no need for learning. To make a prediction on any instance, it first computes the K number of nearest neighbors to that instance. Then it assigned the class to that instance which is the most common class assigned to their neighbors. The optimal choice to determine the value of K is to evaluate the performance of the model on various values of K before settling the value on one. It calculates Euclid's distance on the basis of all attributes of the instances. Sometimes not all of the attributes do not contribute to the classification of the instances. That's why it misleads the results in these cases.

Support Vector Machine uses classical learning approaches which follow the empirical risk minimization (ERM) principal. According to this principal, learning is based on minimizing the error on the training dataset [11]. On the other hand, the SVM follows the structural risk minimization (SRM) principle rooted in the statistical learning theory which gives better generalization abilities to the SVM [12]. According to this principal it minimizes the upper bound of the generalization error.
SVM can handle very large feature spaces because the dimension of classified vectors does not influence the performance of SVM [13]. The performance of SVM in various classification task is given by Cristianini, Shawe-Taylor [14] as in equation 2

$$y_i f(x_i) = y_i(w^T x_i + b) >= 1 \qquad (2)$$

If there is an M number of input samples, then w is a M -dimensional vector and b is scalar. The vectors w and scalar b are used to define the position of separating hyperplanes. f(x) is a separating hyperplane which is used to classify the input data. It performs better over other classifiers if we have unbalanced data or we have a huge difference in the number of positive and negative events. It represents a model using few examples.

Naive Bayes classifier works on a very simplified Bayesian probability model [15]. In the Bayesian probability model, the probability model of one attribute does not affect the probability of the other. For n number of attributes in a dataset, the naive Bayes classifier makes 2n! independent assumption. The probability that an input sample x with attributes vector $< x_1, x_2, .., x_n >$ belongs to class y1 is given by Bayes theorem as in equation 3.

$$P(y_1 |x) = P(x|y_1)P(y_1)/P(x) \qquad (3)$$

where $P(y_1|x)$ is the posterior probability and $P(y_1)$ is prior probability. If this conditional independent assumption holds true, then naive Bayes classifiers perform well and learn rapidly in various supervised classification problems. It is computationally fast and performs well on high dimensional data [16].

Artificial Neural Network is the simulation of real-world neural networks found in animal brains. Scientists are approaching the work and capabilities of neurons found in the animal brain. Neural networks have the capabilities of learning based on the result from experience or training.
An artificial neural network is a highly connected array of elementary processors called nodes or neurons. Multi-layer perceptron is the most widely used model which consists of one input layer, one or more hidden layers and one output layer. Except for the input layer, each neuron receives signals from the neurons of the previous layer. The neuron then produces its output signal by passing the summed signal through an activation function as defined by Sobajic & Pao [17].

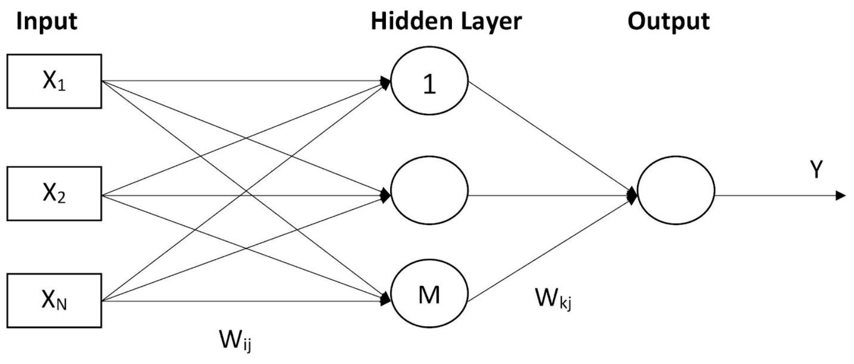

Figure 3: A typical multi-layer feed-forward neural network architecture

ANN faces an issue of overfitting the data and sometimes performs poorly on new data. To overcome this, cross validation can be utilized for the early stopping of gradient descent learning or introducing regularization in the hypothetical function and loss function. Fig 3 represents the architecture of a neural network containing input, one hidden and an output layer.

Logistic Regression contains a dummy variable which is the dependent variable. It is used when the dependent variable is a dichotomous variable and the independent variables can be in continuous, categorical or both. Logistic regression does not assume that the relationship between independent variables and the dependent variable is linear as illustrated by Midi, Sarkar, & Rana [18]. So a hypothetical function for logistic regression is slightly modified by applying an activation function on it which is given in equation 4 and 5. As to convert

$$z = wx + b \tag{4}$$

$$h(x) = f(z) \tag{5}$$

here w is weights, x is input, b is bias, h(x) is the hypothetical function for logistic regression and f is the activation function. We choose activation functions according to the range of our dichotomous dependent variable. Unlike linear regression, binary logistic regression uses binary cross entropy function for calculating the loss as we have discrete values in our output, mean squared error can’t be used to calculate the loss. Due to its linear decision surface and limited hypothesis space. It can’t solve nonlinear problems.

Decision Tree classifier may follow any approaches like table look-up rules, decision table conversion to optimal decision tree, multistage decision making and sequential approaches. However, the basic idea behind decision trees is to break up a complex decision into a union of simpler decisions and determine the best decision using a greedy approach to be on the safe side [19]. It follows the tree-like structure in which each internal node represents a test on an attribute where the branch represents the outcome of the test. A Decision taken for a class label taken after computing all attributes represented by a leaf node. The criterion for taking decision on each internal node may be based on Gini index or Information Gain. As for categorical attributes can use information gain and if attributes are continuous, can utilize Gini index. Entropy is given by equation 6. To avoid overfitting in decision trees, the growth of the tree can be stopped earlier, before it classifies the training data perfectly or we can also prune the tree even without stopping the growth of the tree earlier.

$$H(x) = -p(x)\log(p(x)) \quad (6)$$

Gradient Boosting is an ensemble of various models which help in reducing the factors like bias and variance during predictions. It uses a prediction model which is produced in the form of an ensemble of weak prediction models. It uses a prediction model which is produced in the form of an ensemble of weak prediction models. It usually converts weak learners into strong learners. As in case of Gradient Boosting, it first trains a tree in which each observation is assigned an equal weight. Increase the weights of those observations which are difficult to classify and decrease weights for those which are easy to classify after the evaluation of the first tree. Furthermore, the tree is therefore grown on this new weighted data. After building these 2 trees, it computes the classification error from this new ensemble model and grows a third tree to predict the revised residuals. This process of growing trees repeats for a specified number of iterations. One of the biggest advantages of Gradient Boosting is that it offers us to optimize a user defined cost function, instead of less control loss functions.

Random Forest is based on the Decision Tree algorithm. As in a decision tree, the result data set is split as different as it can be from each other into tree branches and tries to bunch up as a similar result data set it can be. In Decision Tree there's only one tree but in Random Forest Tree there are groups of decision trees to make a forest which are uncorrelated to each other but will perform the same operation on individual models, it increases the probability of accurate result. Random forest can perform comprehensive calculations. When the number of observations are high with any number of variables, random forest generates multiple decision tree models with diverse samples and initial variables. It is magnificent in accuracy. Random forest can be used for future calculation on different data sets to obtain results. For the information regarding relation between the variables and classification computed by the prototypes. Random forest comparatively from other machine learning algorithms is less impacted by noise. It is more reliable even if a new data point is added into the data set, it will not affect the overall algorithm since the new data may impact one or two trees but it is very unlikely for it to impact all the trees. Random forest can handle missing values itself.

Experiments and Results

System Specifications of the machine on which the model training and testing was performed on the GPU with the following specifications given in Table 4:

Table 4: GPU Specification

Model name	Intel(R) Xeon(R) Gold 6140 CPU @ 2.30GHz
Architecture	x86_64
CPU(s)	72
CPU MHz	1000.713
CPU max MHz	3700.0000
CPU min MHz	1000.0000
RAM	128 GB
GPU	Nvidia Quadro P5000
GPU Memory	16 GB

Training Configuration constitutes trained data which is on 2,40,000 classifiers events out of 2,50,000 and it is tested on 10,000 events.

For k-nearest neighbor classifiers, as it is a lazy learning algorithm. It learns only during prediction. At the small values of k (neighbors), the model has a variance problem as it performs well on the training set but did not do well on the test set. During large k it has a bias problem as it did not even fit the training set well. When the value of K is equal to 7, it gave the best result.

In case of Support Vector Machine, radial basis function used as kernel function and used scale coefficient for kernel function which defines the influence of a single input pattern.

As dealing with continuous data it can be concluded that features of events are distributed according to the Normal or Gaussian distribution. So Gaussian naive Bayes classifiers can be used for the experiment.

In the case of neural networks, a rectified linear unit activation function is used in the hidden layer. The reason behind using the rectified linear unit activation function in the hidden layer is to let the gradient be non-zero and recoverable during training. Sigmoid activation function in the output layer neuron because we have to predict the probability as output. Stochastic gradient descent algorithm is used to learn the weights of neurons in which we set its batch size to 32. Generally, it is noisier than simple gradient descent because it takes a higher number of iterations to reach the global minima. But it is computationally less expensive than simple gradient descent. It does not feed all training examples in once to calculate the cost gradient on every iteration. So, in the case of large datasets as in this paper, it is better to use stochastic gradient descent for efficient optimization. The learning rate is initialized with 0.02 with a decay of e^{-6} to update the learning rate in each epoch. Several experiments were performed by changing the number of neurons in the hidden layer and evaluating their performances on the training set and test set. But the best results were achieved when the numbers of neurons in the hidden layer were chosen same as in the input layer.

In logistic regression when it started decreasing regularization strength by increasing the value of inverse regularization strength C, the classifier performs well. At C is equal to 200 it gave the best result on training and test set.

As in case of Decision tree and random forest classifier, this dataset has continuous data in independent variables Gini criterion is utilized for splitting the attributes at each internal node. This paper determines that Decision tree classifier, Gradient Boosting and Random Forest performed exceptionally well, as it achieved 100% accuracy on training as well as test set. Performance on the train and test set of all classifiers is listed in Table 5.

Table 5: Performance Measure

Classifier	Training accuracy	Test accuracy
K-Nearest Neighbor	92.58%	90.57%
Support Vector Machine	97.05%	96.51%
Naive Bayes	99.28%	99.26%
Artificial Neural Network	99.83%	99.81%
Logistic Regression	99.99%	100%
Decision Tree	100%	100%
Gradient Boosting	100%	100%
Random Forest	100%	100%

Abbreviation which we used for the classifiers are given below:

KNN	K Nearest Neighbor
SVM	Support Vector Machine
NB	Naive Bayes
ANN	Artificial Neural Network
LR	Logistic Regression
DT	Decision Tree
GB	Gradient Boosting
RF	Random Forest (Abbreviation)

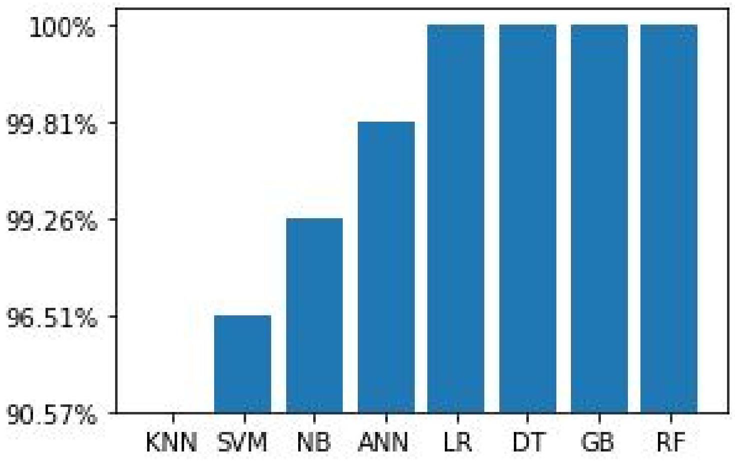

Figure 4: Test accuracy

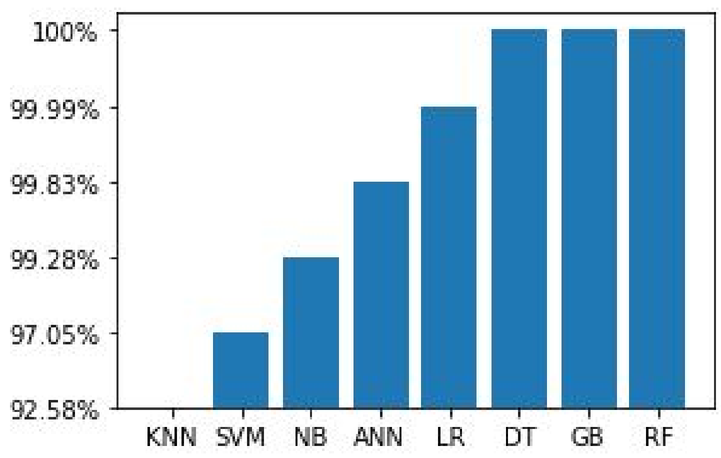

Figure 5: Training accuracy

Fig. 5. and Fig. 4. show performances of all classifiers on the training set and test set.

Time Analysis

In this experiment time analysis is performed on GPU based computation. Each classifier algorithm 5 times on training and test data and calculated the average value. The computation time taken by GPU over training data and test data for each classifier are listed in Table 6 and Fig. 6 and Fig. 7 shows the graphical comparison of time analysis for all classifiers. In case of K-nearest neighbor, its training time is very less than the testing time due to its lazy algorithm-based principal, as it learns only during predictions. Naive Bayes classifier showed us the most optimizing behavior in contrast to training, but it took more time to predict than that of logistic regression and decision trees. logistic regression after well tuning of hyperparameters gave us impressive results during predictions, as its testing time is lowest in all classifiers. Time analysis showed that naive Bayes classifiers performed well on training as well as on test data for the time point of view.

Table 6: GPU based computation

Classifier	Training time	Testing time
K-Nearest neighbor	3.225s	7.964s
Support Vector Machine	1008.156s	9.718s
Naive Bayes	0.131s	0.006s
Artificial Neural Network	3589.567s	0.268s
Logistic Regression	8.456s	0.001s
Decision Tree	0.554s	0.003s
Gradient Boosting	20.164s	0.011s
Random Forest	4.760s	0.011s

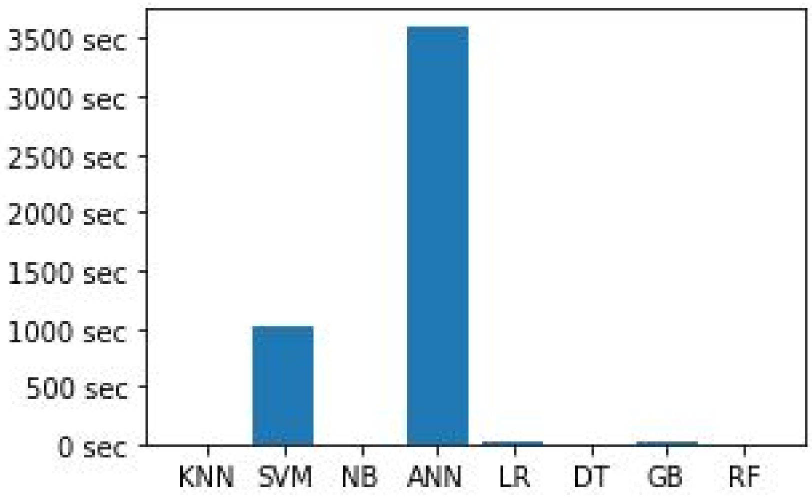

Figure 6: Training Time

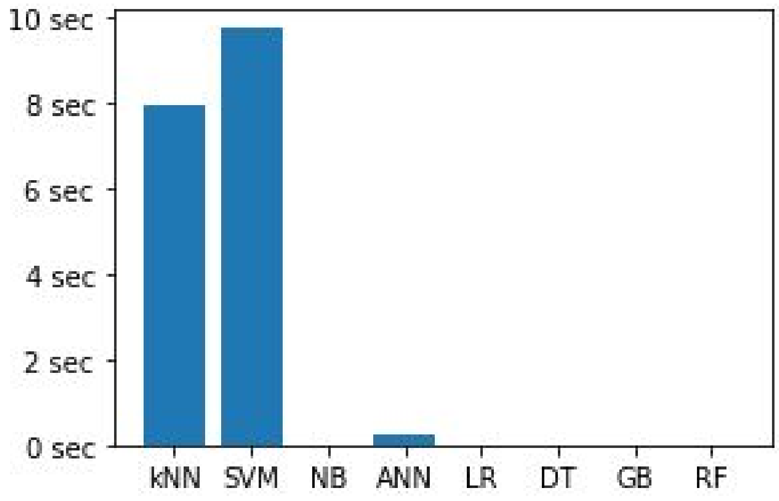

Figure 7: Testing Time

Future Scope
State-of-art ML algorithms, as neural networks, have pushed the limits of the LHC to discover new particles. Nevertheless, many other techniques and methods from the machine learning field are still waiting to be applied in particle physics. One of these techniques is ensemble learning, where many ML algorithms are trained to attack the same type of problem. Based on these results, maybe these algorithms might not perform well in general but assembling several algorithms can be performed to produce better results. In this case, Random Forest and Gradient Boosting were better than other algorithms but this was on pre-recorded data. However, we can generate more potentially powerful algorithms by ensemble technique which can be used to analyze the data and will help in online filtering of streaming detector measurements.

Conclusion

Machine learning has led to significant advances in many fields such as data analysis, web search, photo tagging and spam detector, etc. It is clearly useful for a wide range of applications, including a host of applications in the natural sciences. The problems in high-energy physics are particularly suitable for machine learning, having large data sets with complicated underlying structure. Results showed that machine learning can be used for providing a powerful and practical approach to analyzing particle collider data. On the basis of results, it clearly specifies that the most accurate classifiers are Logistic Regression, Gradient Boosting and Random Forest but there are some drawbacks when it comes to time analysis of the above classifiers. Turns out Naïve Bayes is the most optimized of them all but it took more time than Logistic Regression and Decision Tree. Therefore, we can say that Decision Tree is the most efficient classifier when it comes to inspecting such large data sets with accuracy and computational time.

References

[1] New results indicate that new particle is a higgs boson. (2013, March). Retrieved 2013-03-14, from https://home.cern/news/news/physics/new-results-indicate-new-particle-higgs-boson

[2] The large hadron collider. (n.d.). Retrieved from https://home.cern/science/accelerators/large-hadron-collider

[3] Atlas. (n.d.). Retrieved from https://home.cern/science/experiments/atlas

[4] Whiteson, S., Whiteson, D. (2009). Machine learning for event selection in high energy physics. Engineering Applications of Artificial Intelligence, 22(8), 1203–1217.

[5] Denby, B. (1999). Neural networks in high energy physics: a ten-year perspective. Computer Physics Communications, 119(2-3), 219–231.

[6] Baldi, P., Sadowski, P., Whiteson, D. (2014). Searching for exotic particles in high-energy physics with deep learning. Nature communications, 5, 4308.

[7] Alves, A. (2017). Stacking machine learning classifiers to identify higgs bosons at the lhc. Journal of Instrumentation, 12(05), T05005.

[8] Cern open data portal. (2014). Retrieved from http://opendata.cern.ch/

[9] Higgs boson training data. (2014). Retrieved from https://www.kaggle.com/c/higgs-boson/data

[10] Peterson, L. E. (2009). K-nearest neighbor. Scholarpedia, 4(2), 1883.

[11] Boser, B. E., Guyon, I. M., Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on computational learning theory (pp. 144–152).

[12] Vapnik, V. (2013). The nature of statistical learning theory. Springer science & business media.

[13] Cortes, C., Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273–297.

[14] Cristianini, N., Shawe-Taylor, J., et al. (2000). An introduction to support vector machines and other kernel-based learning methods. Cambridge university press.

[15] Wafa'S, A.-S., Naoum, R. (2009). Development of genetic-based machine learning for network intrusion detection. World Academy of Science, Engineering and Technology, 55, 20–24.

[16] Dimitoglou, G., Adams, J.A., Jim,C.M. (2012). Comparison of the c4.5 and a naïve bayes classifier for the prediction of lung cancer survivability. arXiv preprint arXiv:1206.1121.

[17] Sobajic, D. J., Pao, Y.-H. (1989). Artificial neural-net based dynamic security assessment for electric power systems. IEEE Transactions on Power Systems, 4(1), 220–228.

[18] Midi, H., Sarkar, S. K., Rana, S. (2010). Collinearity diagnostics of binary logistic regression model. Journal of Interdisciplinary Mathematics, 13(3), 253–267.

[19] Dattatreya, G., Kanal, L.N. (1985). Decision trees in pattern recognition. University of Maryland. ComputerScience.