International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 12 | Dec 2021 www.irjet.net p-ISSN: 2395-0072

Higgs Boson Discovery using Machine Learning

Vinay Chaukate[1], Abhishek Yadav[2], Ravi Jogdand[3], Jayesh Vanmare[4], Prof. Ankur Ganorkar[5]

[1]Vinay Chaukate, Elec. and Telecomm. Engineering, Rajiv Gandhi Institute and Technology, Mumbai, India
[2]Abhishek Yadav, Elec. and Telecomm. Engineering, Rajiv Gandhi Institute and Technology, Mumbai, India
[3]Ravi Jogdand, Elec. and Telecomm. Engineering, Rajiv Gandhi Institute and Technology, Mumbai, India
[4]Jayesh Vanmare, Elec. and Telecom. Engineering, Rajiv Gandhi Institute and Technology, Mumbai, India
[5]Prof. Ankur Ganorkar, Rajiv Gandhi Institute and Technology, Mumbai, India

---

Abstract - *The Higgs Boson is a fundamental particle that imparts mass to all matter in nature. The search for the Higgs boson is a major undertaking in particle physics. The Higgs Boson Classification Problem is proposed to be solved using machine learning (ML) approaches such as long-short-term memory (LSTM) and decision trees in this research (DT). The accuracy and AUC measures of those machine learning approaches are compared. We employ a huge dataset as the Higgs Boson, obtained from the public site UCI, and a Higgs dataset collected from the Kaggle site during the experimentation stage.*

Key Words: **Dataset, Workflow of Machine Learning, Deployment.**

1.INTRODUCTION

The Higgs Boson's discovery is a big issue for the field of High Energy Physics. To overcome the challenge of signal separation from background events, machine learning algorithms are presented. The signal is the decay of exotic particles, which creates a zone in feature space that is not explained by background processes. The background is made up of dissolving particles that have been found in prior tests. Particle physics has progressed in several ways that are pertinent to the quest for the Higgs Boson. ATLAS detector researchers evaluated the Standard Model predictions as part of one of the key experiments at the European Organization for Nuclear Research's (CERN) Large Hadron Collider (LHC). Six algorithms were used in the Higgs Boson Machine Learning Challenge, which was held in 2014. Support Vector, Machine Affinity Propagation, Random Forest, Decision Tree, K-Nearest Neighbors, K Means Clustering, Support Vector, Machine Affinity Propagation They used Higgs datasets to test the Deep Networks Classifier. Alves employed Stacking Machine Learning classifiers in a multivariate statistical analysis (MVA) to find Higgs Bosons at the LHC, outperforming Boosted Decision Drees and Deep Neural Network applications in particle physics. The following study suggests using machine learning (ML) approaches to address the Higgs Boson categorization problem: long short-term memory (LSTM) and decision tree (DT). Using the Higgs dataset UCI and the Higgs dataset Kaggle, we compare the accuracy of those ML algorithms.

1.1 Working Principle

Several machine learning methods were used to create classifiers before classifying the data samples. These models are expected to be either a signal or a background. Training data was used to determine the component models. An integrated model was created to boost performance.

High-energy physicists employ a variety of machine learning approaches to optimise and analyse the selection zone that generates these signal occurrences. Simulated signal and background events are used to train classifiers, which are given a weight to account for the difference between the event's prior probability and the simulator's probability.

1.2 Overview

Data pre-processing, data purification, feature identification, and feature engineering will also be discussed, as well as how they affect Machine Learning Model Performance. We'll also go through a few pre-modeling procedures that can help the model run faster. For deployment, we created a webapp. Python Libraries that would be need to achieve the task:
1.NumPy
2.Pandas
3.TensorFlow
4.Sci-kitLearn
5. Matplotlib

2. DATASET

In an eight-dimensional feature space, the data comprises of simulated signal and background events. Each event data point is allocated an ID and a weight, as previously stated. The 8 attributes were actual values and comprised estimated particle mass, invariant mass of hadronic tau and lepton, vector sum of the transverse momentum of hadronic tau, centrality of azimuthal angle, pseudo-rapidity of the leptons, number of jets and their properties, and so on. A total of 250,000 incidents were included in the data. Weights were not included with the test results. Each training data event was labelled with one of two labels: "s" for signal and "b" for background.

	DER_mass_MMC	DER_mass_transverse_net_lep	DER_mass_vis	DER_pt_h	DER_deltaeta_jet_jet	DER_mass_jet_jet	DER_prodeta_jet_jet	Label
0	138.470	51.655	97.827	27.980	0.91	124.711	2.666	s
1	160.937	68.768	103.235	48.146	-999.00	-999.000	-999.000	b
2	-999.000	162.172	125.953	35.635	-999.00	-999.000	-999.000	b
3	143.905	81.417	80.943	0.414	-999.00	-999.000	-999.000	b
4	175.864	16.915	134.805	16.405	-999.00	-999.000	-999.000	b
...	...	...	...	...	...	...	...	...
249995	-999.000	71.989	36.548	5.042	-999.00	-999.000	-999.000	b
249996	-999.000	58.179	68.083	22.439	-999.00	-999.000	-999.000	b
249997	105.457	60.526	75.839	39.757	-999.00	-999.000	-999.000	s
249998	94.951	19.362	68.812	13.504	-999.00	-999.000	-999.000	b
249999	-999.000	72.756	70.831	7.479	-999.00	-999.000	-999.000	b

250000 rows × 8 columns

Dataset

3. WORKFLOW OF MACHINE LEARNING

Understanding the machine learning workflow

We can define the machine learning workflow in 3 stages.

- Gathering data
- Data pre-processing
- Researching the model that will be best for the type of data
- Training and testing the model
- Evaluation

What is the machine learning Model?

A machine learning model is nothing more than a piece of code that has been trained with data by an engineer or data scientist. So, if you feed the model garbage, you'll receive garbage back, i.e., the trained model will make erroneous or incorrect predictions.

1. Gathering Data

The method for acquiring data depends on the sort of project we want to create. For example, if we want to create an ML project that uses real-time data, we can create an IoT system that uses data from various sensors. The data set can come from a variety of places, including a file, a database, a sensor, and many other places, but it cannot be utilised immediately for analysis since there may be a lot of missing data, extremely big values, disorganised text data, or noisy data. As a result, Data Preparation is completed to address this issue. We can also make use of various free data sets available on the internet. The most popular repositories for creating Machine Learning models are Kaggle and the UCI Machine Learning Repository. Kaggle is one of the most popular websites for practising machine learning algorithms. They also hold events in which users can compete and put their machine learning skills to the test.

2. Data pre-processing

One of the most important steps in machine learning is data pre-processing. It is the most crucial step in improving the accuracy of machine learning models. There is an 80/20 rule in machine learning. Every data scientist should devote 80% of their time to data pre-processing and 20% of their time to actual analysis.

What is data pre-processing?

Data pre-processing is the process of cleaning raw data, which is data that has been obtained in the real world and converted into a clean data set. In other words, anytime data is received from many sources, it is collected in a raw format, which makes analysis impossible. As a result, specific processes are taken to convert the data into a smaller, clean data set, which is referred to as data pre-processing.

Why do we need it?

Data pre-processing, as we all know, is the process of converting raw data into clean data that can be utilised to train the model. To get decent outcomes from the applied model in machine learning and deep learning projects, we surely require data pre-processing.

Most of the real-world data is messy, some of these types of data are:

1. **Missing data:** Missing data can be found when it is not continuously created or due to technical issues in the application.

2. **Noisy data**: This type of data is also called outliners; this can occur due to human errors (human manually gathering the data) or some technical problem of the device at the time of collection of data.

3. **Inconsistent data**: This type of data might be collected due to human errors (mistakes with the name or values) or duplication of data.

3. Researching the model that will be best for the type of data

Our main goal is to train the best performing model possible, using the pre-processed data.

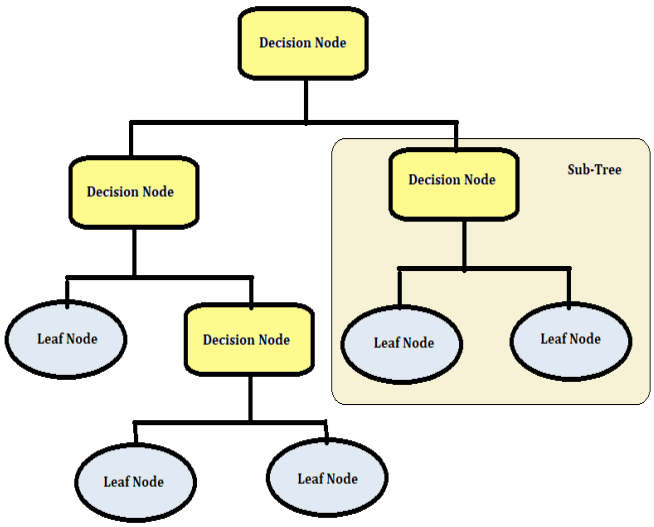
Supervised Learning

In Supervised learning, an AI system is supplied with data which is labelled, which means that each data tagged with the right label. The supervised learning is classified into 2 more categories which are “Classification” and “Regression”.

In the classification and regression phase, we used a few approaches.

Decision Tree Algorithm

A decision tree is a tree structure that looks like a flowchart, with an internal node representing a feature (or attribute), a branch representing a decision rule, and each leaf node representing the outcome. The root node is the topmost node in a decision tree. It learns to partition based on the value of an attribute. Recursive partitioning is a method of partitioning the tree in a recursive manner. This flowchart-like structure assists you in making decisions. It's a flowchart diagram-style depiction that closely resembles human thinking. As a result, decision trees are simple to comprehend and interpret.



The decision tree is a non-parametric or distribution-free strategy that does not rely on probability distribution assumptions. With good accuracy, decision trees can handle high-dimensional data.

Linear Regression

A supervised classification algorithm, logistic regression is. For a given collection of features (or inputs), X, the target variable (or output), y, can only take discrete values in a classification issue. Logistic regression, contrary to popular assumption, is a regression model. The model creates a regression model to forecast the likelihood that a given data entry belongs to the "1" category. Logistic regression models the data using the sigmoid function, just like linear regression assumes that the data follows a linear distribution. When a decision criterion is introduced, logistic regression transforms into a classification procedure. Setting the threshold value is a crucial part of Logistic regression, and it is determined by the classification problem.

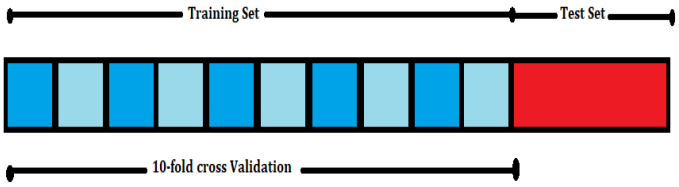
Long Short-Term Memory (LSTM):

A recurrent neural network is a type of long short-term memory. The output of the previous step is used as input in the current step in RNN. Hoch Reiter and Schmid Huber created LSTM. It addressed the issue of RNN long-term dependency, in which the RNN is unable to predict words stored in long-term memory but can make more accurate predictions based on current data. RNN does not provide an efficient performance as the gap length rises. By default, the LSTM can keep the information for a long time. It is used for time-series data processing, prediction, and classification. The LSTM has a chain structure that consists of four neural networks and several memory blocks known as cells.

4. Training and testing the model on data

To begin training a model, we divide it into three sections: 'Training data," Validation data,' and 'Testing data.'

You use a 'training data set' to train the classifier, a 'validation set' to modify the parameters, and a 'unseen test data set' to test the classifier's performance. It's worth noting that only the training and/or validation sets are available to the classifier during training. The test data set must not be used when the classifier is being trained. The test set will only be available while the classifier is being tested.



The training set is the content that the computer uses to learn how to process information. The training component of machine learning is done through algorithms. A set of data used for learning, or fitting the classifier's parameters.

Set of validations: Cross-validation is a technique used in applied machine learning to estimate a machine learning model's skill on unknown data. To tune the parameters of a classifier, a collection of unseen data is used from the training data.

A test set is a collection of previously unseen data used solely to evaluate the performance of a fully stated classifier.

We can begin the training procedure once the data has been separated into the three segments specified.

A training set is used to develop a model in a data collection, whereas a test (or validation) set is used to test the model. The test (validation) set excludes data points from the training set. In each iteration, a data set is usually divided into a training set, a validation set (some people call it a 'test set' instead), or a training set, a validation set, and a test set.

Any of the models we choose in step 3/ point 3 are used in the model. We can use the same trained model to forecast

using the testing data, i.e. the unseen data, once the model has been trained. After that, we can create a confusion matrix, which will tell us how successfully our model has been trained. True positives, True Negatives, False Positives, and False Negatives are the four parameters of a confusion matrix. To create a more accurate model, we desire to get more values in the True negatives and True positives. The number of classes has no bearing on the size of the Confusion matrix.

N=165	Predicted No	Predicted No
Actual: No	50	10
Actual: Yes	5	100

True positives: These are instances where we predicted TRUE and our anticipated output was accurate.

True negatives: We predicted FALSE and were correct in our prediction.

False positives occur when we expect TRUE but the actual outcome is FALSE.

False negatives occur when we expect FALSE while the actual projected output is TRUE.

The confusion matrix can also be used to determine the model's correctness.

Accuracy = (True Positives +True Negatives) / (Total number of classes)

i.e. for the above example:

Accuracy = (100 + 50) / 165 = 0.9090 (90.9% accuracy)

5. Evaluation

Model evaluation is an important step in the creation of a model. It aids in the selection of the best model to represent our data and the prediction of how well the chosen model will perform in the future. We could tweak the model's hyper-parameters to improve accuracy, as well as look at the confusion matrix to see if we can increase the number of true positives and true negatives.

4. DEPLOYMENT

We designed a webapp for deployment. Using streamlit and Heroku, we deployed the trained model. We used streamlit to create a frontend, which we then deployed to the Heroku server. We used Mauna to deploy our GitHub repository to Heroku; the deployment option was Manually.

Streamlit is a Python-based app framework for deploying machine learning apps. It's a free and open-source framework that's similar to R's Shiny package. Heroku is a cloud-based platform-as-a-service (PaaS) that facilitates the deployment and management of apps written in a variety of programming languages.

Heroku lets you deploy, run and manage applications written in Ruby, Node.js, Java, Python, Clojure, Scala, Go and PHP.

Heroku allows you to launch, execute, and manage Ruby, Node.js, Java, Python, Clojure, Scala, Go, and PHP applications. An application is made up of source code written in one of these languages, maybe a framework, and a dependency description that tells a build system which other dependencies are required to create and run the programmed. The source code for your project, along with the dependency file, should offer enough information for the Heroku platform to build and run your app.

Many developers use Git to manage and version source code because it is a powerful, distributed version management system. Git is the primary method for deploying applications on the Heroku platform. When you create an app on Heroku, it creates a new Git remote, usually named Heroku, that is linked to your app's local Git repository.

Deployment, then, is the process of moving your programmed from your local system to Heroku, and Heroku offers numerous options for doing so.

5. CONCLUSIONS

Even in the Higgs field, it's difficult to find signals from the background. To overcome the challenge of signal separation from background events, machine learning algorithms are presented. We used a machine learning model to detect Higgs boson signals or background with high accuracy, and because this machine learning model is fast, it saves us a lot of time. As far as Higgs boson practical discovery, The LSTM runs well with accuracy achieved 79%. It works better than Decision Tree (72%) and Linear Regression (67%).

Future Scope

More higgs data will be available in the future. We can easily determine whether they are signals or background using machine learning. Using several models, we can improve accuracy. We create programmed that are simple to use for everybody.

ACKNOWLEDGEMENT

It has been a great opportunity to gain lots of experience in real time projects, followed by the knowledge of how to actually design and analyze real projects. For that we want to thank all the people who made it possible for students like

us. Special thanks to the graduation Project Unit for the efforts they did to provide us with all useful information and making the path clear for the students to implement all the education periods in real-time project design and analysis. Furthermore, we all the professors and visiting industry for the interesting lectures they presented which had great benefit for all of us. We would like to express our deepest gratitude to our graduation project supervisor Prof. Ankur Ganorkar for his patience and guidance along the semester. In addition, we would like to express our sincere appreciations to ours.

REFERENCES

[1] Cecilia Tosciri, "Machine Learning Applications and Observation of Higgs Boson Decays into a Pair of Bottom Quarks with the ATLAS Detector", in 2020.

[2] Present by Mourad Azharia, Abdallah Abardab, Badia Ettakia, c, Jamal Zerouaouia, Mohamed Dakkon "Higgs Boson Discovery using Machine Learning Methods with Pyspark", in 2020.

[3] Present by Anton Apostolatos and Leonard Bronner "Identifying the Higgs Boson with Convolutional Neural Networks", in 2017.

[4] Present by S. Raza Ahmad "Higgs Boson Machine Learning Challenge", in 2014.

[5] Present by Dennise Silverman "Importance of The Higgs Boson Discovery", in 2012.

https://sites.uci.edu/energyobserver/2012/11/25/importance-of-the-higgs-boson-discovery/

[6] Present by Alexander Radovic, Mike Williams, David Rousseau, Michael Kagan, Daniele Bonacorsi, Alexander Himmel, Adam Aurisano, Kazuhiro Terao & Taritree wongjirad "Machine learning at the energy and intensity frontiers of particle physics", in 2018.

[7] Present by Jocelyn Perez, Ravi Ponmalai, Alex Silver, Dacoda Strack, "Higgs Boson Machine Learning Challenge", in 2014.

[8] Present by Vishal S. Ngairangbam, Akanksha Bhardwaj, Partha Konar, Aruna Kumar Nayak "Invisible Higgs search through vector boson fusion: a deep learning approach", in 2020.

[9] Present by G. Rajasekaran "THE STANDARD MODEL AND THE HIGGS BOSON", in 2010.

[10] 10] Present by P. Baldi, P. Sadowski & D. Whiteson, "Searching for exotic particles in high-energy physics with deep learning", in 2014.