Cricket Bat Swing Detection Based on 6-Axis IMU Dataset

[1] Piyush Agarwal, [2] Shalette Natasha D'Souza
Department of Computer Science & Engineering
BMS Institute of Technology & Management
Bengaluru, Karnataka, India
Email : [1]1by16cs059@bmsit.in, [2]1by16cs074@bmsit.in

***Abstract*—Cricket is one of the most popular sports in the world and many students aspire to take cricket professionally. Technology can be a friend to provide aid to those students who can't take expensive coachings or professional help. To minimize this gap a lot of organizations are working on solutions that can analyze and recommend improvements on the user's play. One such approach is to use IoT devices to capture orientation, position, velocity data, and build a recommendation system. But the basic implementation at least needs to first detect a swing made by the batsman so that later analysis can be done. For this, we need to find a suitable algorithm that can perform real-time classifications accurately and efficiently.**

***Index Terms*—Cricket, Bat, Swing Detection, Machine learning, Spark, IMU, BNO055, Accelerometer, Gyroscope**

I. Introduction

The Internet of Things (IoT) has been joining the digital and physical worlds for years. Today, everything we think of as "smart" technology can be attributed to IoT. If you're developing something smart in 2018, you're going to connect it to other smart devices. IoT developers are not only becoming more in demand; they're helping futureproof the things we use every day in areas as basic yet essential as lighting, heating and cooling, kitchen appliances, etc. Then there are more involved and seemingly futuristic use cases, like robotics, autonomous machines and medical devices. Wearable tech, like Fitbits, have been helping regular and aspiring athletes for years now. Wearables can tell users a lot about their physical experience, including heart rate, distance traveled, calories burned, sleep quality, and more. It makes sense, then, that this technology could move into different wearable items to tell professional athletes about their performance.

Some sporting teams have dabbled in adding AI-run chatbots into their technology mix to give fans a better experience, even away from the arena. Fans on the go, for instance, could message the chatbot and ask for statistics or updates, making it easier to follow their favorite teams. The cricket industry is now also adapting to the tremendous benefits of the iot in recent days and one such place for innovation has been coaching players from distance by the use of iot devices which provides useable, real-time and instant 360-degree batting performance data analytics to players and coaches through iOS and Android-enabled mobile app. Many companies have come up with ideas to make it as efficient and reliable as possible but main challenges arise at the very beginning and that is the analysis of a swing from all the moments that a player does.

II. Objective

Our objective is to accurately classify a bat swing from all the moments done by a cricket player with the bat based on the data generated by 6-axis IMU device such as BNO055 in least possible training and testing time.

III. Dataset

A. Dataset Collection

The dataset is collected from the sessions of 5 professional students coached under prestigious cricket academies. During the session the sensor was mounted on the bats of each player and the data is recorded and labeled on hand as the player played. The data is cleansed and reformatted for the use after the recording.

B. Dataset Composition

The dataset consists of 1278 csv files out of which 806 files contain data that represents a successful professional swing of a bat and 472 files containing the random moments that players may do during their play. Each file contains 100 rows representing data for 2 second of bat moment and 6 columns representing Linear Acceleration and Rotational Velocity in X, Y and Z axes.

IV. Technologies Used

We have used Apache spark for distributed and parallel computing and used its MLlib Library for all machine learning related tasks. The language use is python 3.6 for convenience and does not affect the overall result. The models are trained and tested on google colabs, a free utility provided by google research team for data science related projects. All graphs are made using Matplotlib library.

V. Operational Environment

In the operational environment the batsman will have the sensor placed on the top of the handle so that it doesn't get damaged during gameplay. The sensor on start of the session will start sending velocity and acceleration data to the mobile application via either bluetooth or wifi. The Application will collect and send the data to the server for classification on

session end. The results may comprise additional analysis after swing detection which is shown to the user. A typical session usually has 30 to 50 swings and a player usually goes for 3 to 5 sessions in an hour hence high accuracy and fast response is necessary.

VI. Implementation

A. Decision Tree Classifier

A Decision Tree [1] is a simple representation for classifying examples. It is a Supervised Machine Learning where the data is continuously split according to a certain parameter.

Training Dataset Count	1032
Test Dataset Count	244
Accuracy	0.98770
Random (Chance) Prediction AUROC	0.5
Decision Tree Classifier AUROC	0.9878
Training Time (Sec)	7.98
Testing Time (Sec)	0.16

TABLE I
Result

	Precision	Recall	f1-score	Support
0	0.98	0.99	0.98	84
1	0.99	0.99	0.99	160
Accuracy			0.99	244
Macro Avg.	0.99	0.99	0.99	244
Weighted Avg.	0.99	0.99	0.99	244

TABLE II
Metrices

	True	False
Positive	82	1
Negative	2	158

TABLE III
Confusion Matrix

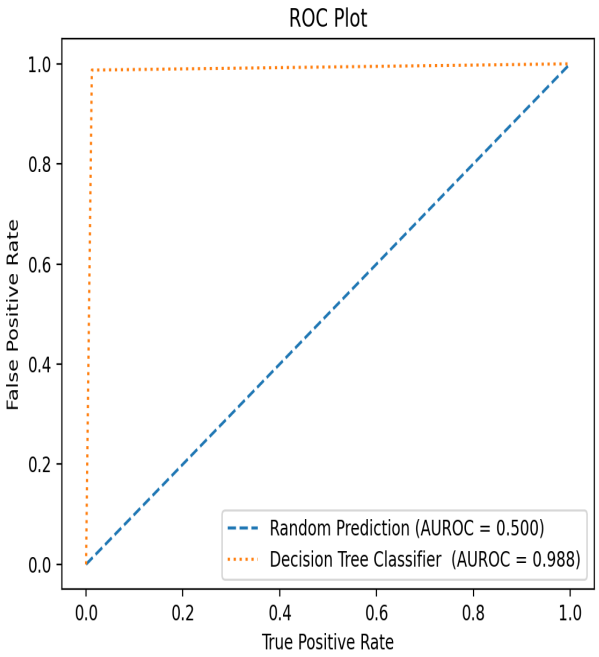

Fig. 1. ROC PLOT

B. Gradient-Boosted Tree Classifier

Gradient boosting [2] is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees.

Training Dataset Count	1032
Test Dataset Count	244
Accuracy	0.98770
Random (Chance) Prediction AUROC	0.5
Decision Tree Classifier AUROC	0.9938
Training Time (Sec)	15.33
Testing Time (Sec)	0.16

TABLE IV
Result

	Precision	Recall	f1-score	Support
0	0.98	0.99	0.98	84
1	0.99	0.99	0.99	160
Accuracy			0.99	244
Macro Avg.	0.99	0.99	0.99	244
Weighted Avg.	0.99	0.99	0.99	244

TABLE V
Metrices

	True	False
Positive	82	1
Negative	2	158

TABLE VI
CONFUSION MATRIX

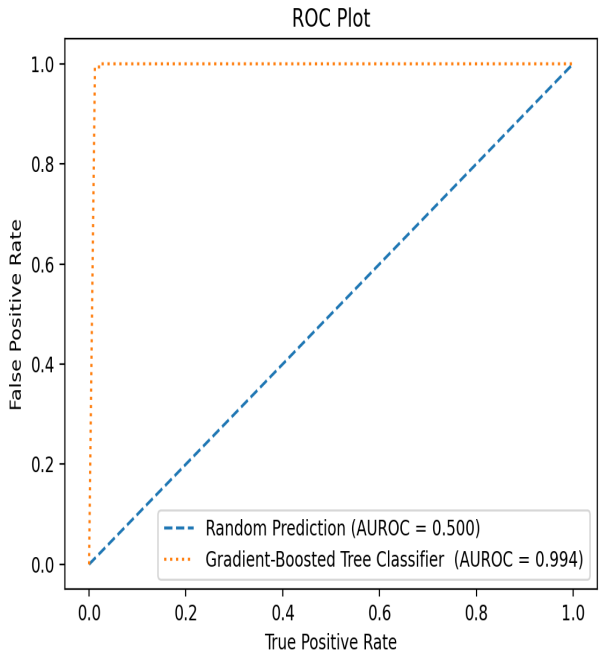

Fig. 2. ROC PLOT

C. Logistic Regression Model

Logistic regression [3] is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression).

Training Dataset Count	1032
Test Dataset Count	244
Accuracy	0.9836
Random (Chance) Prediction AUROC	0.5
Decision Tree Classifier AUROC	0.9982
Training Time (Sec)	4.93
Testing Time (Sec)	0.12

TABLE VII
RESULT

	Precision	Recall	f1-score	Support
0	0.99	0.96	0.98	84
1	0.98	0.99	0.99	160
Accuracy			0.98	244
Macro Avg.	0.98	0.98	0.98	244
Weighted Avg.	0.98	0.98	0.98	244

TABLE VIII
METRICES

	True	False
Positive	81	3
Negative	1	159

TABLE IX
CONFUSION MATRIX

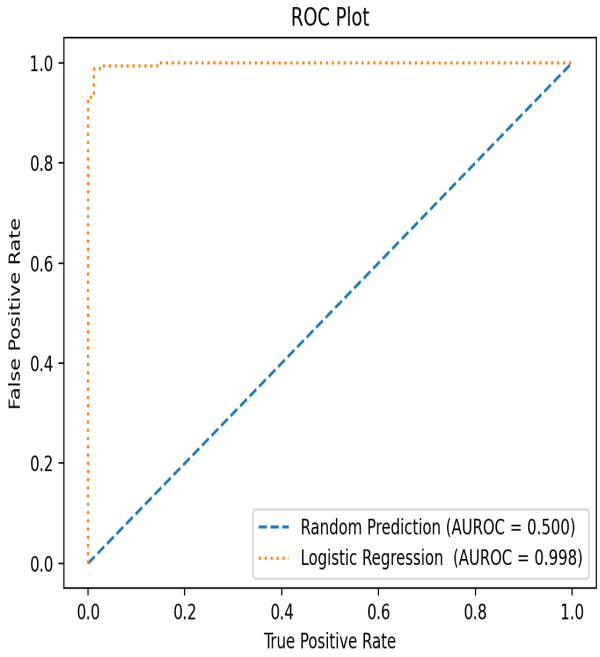

Fig. 3. ROC PLOT

D. Multilayer Perceptron Classifier

A multilayer perceptron (MLP) [4] is a class of feedforward artificial neural network (ANN). The term MLP is used ambiguously, sometimes loosely to refer to any feedforward ANN.An MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training.

Training Dataset Count	1032
Test Dataset Count	244
Accuracy	0.9016
Random (Chance) Prediction AUROC	0.5
Decision Tree Classifier AUROC	0.9676
Training Time (Sec)	16.46
Testing Time (Sec)	0.14

TABLE X
RESULT

	Precision	Recall	f1-score	Support
0	0.88	0.82	0.85	84
1	0.91	0.94	0.93	160
Accuracy			0.90	244
Macro Avg.	0.90	0.88	0.89	244
Weighted Avg.	0.90	0.90	0.90	244

TABLE XI
METRICES

	True	False
Positive	69	15
Negative	9	151

TABLE XII
CONFUSION MATRIX

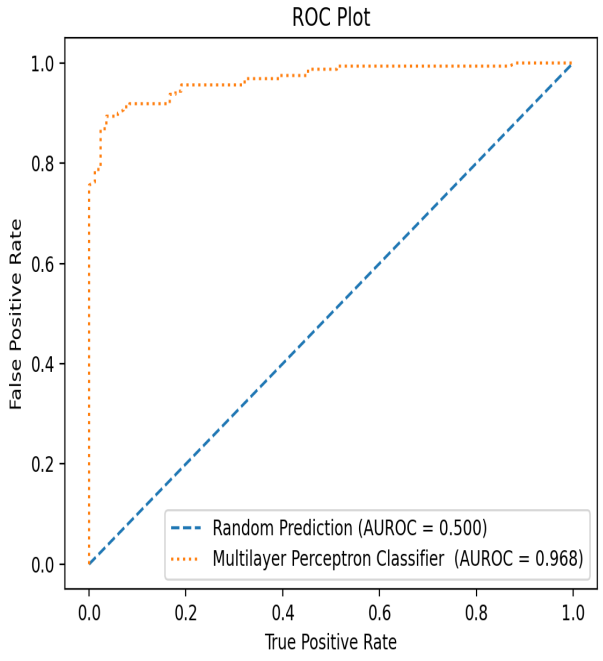

Fig. 4. ROC PLOT

E. Naive Bayes Classifier

In machine learning, naïve Bayes classifiers [5] are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features. They are among the simplest Bayesian network models.[1] But they could be coupled with Kernel density estimation and achieve higher accuracy levels.

Training Dataset Count	1032
Test Dataset Count	244
Accuracy	0.9754
Random (Chance) Prediction AUROC	0.5
Decision Tree Classifier AUROC	1.0
Training Time (Sec)	5.25
Testing Time (Sec)	0.18

TABLE XIII
RESULT

	Precision	Recall	f1-score	Support
0	0.93	1.00	0.97	84
1	1.00	0.96	0.98	160
Accuracy			0.98	244
Macro Avg.	0.97	0.98	0.97	244
Weighted Avg.	0.98	0.98	0.98	244

TABLE XIV
METRICES

	True	False
Positive	84	0
Negative	6	154

TABLE XV
CONFUSION MATRIX

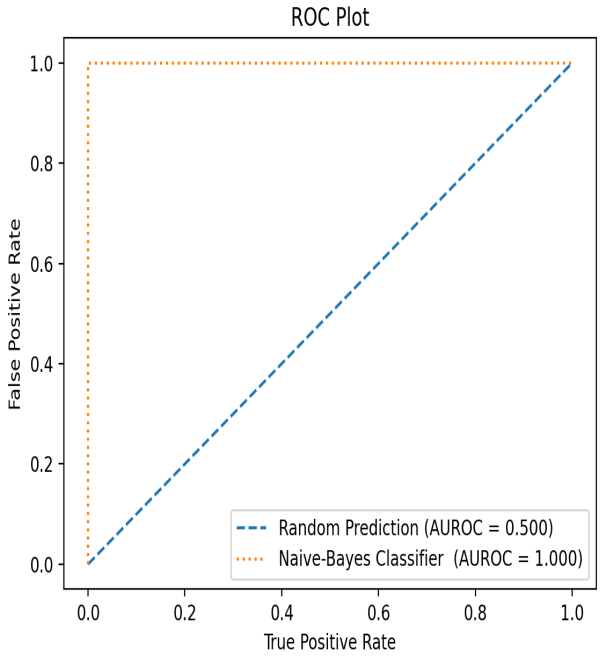

Fig. 5. ROC PLOT

F. Random Forest Classifier

It is an ensemble tree-based learning algorithm. The Random Forest Classifier [6] is a set of decision trees from a randomly selected subset of training set. It aggregates the votes from different decision trees to decide the final class of the test object.

Training Dataset Count	1032
Test Dataset Count	244
Accuracy	0.9959
Random (Chance) Prediction AUROC	0.5
Decision Tree Classifier AUROC	1.0
Training Time (Sec)	9.29
Testing Time (Sec)	0.19

TABLE XVI
RESULT

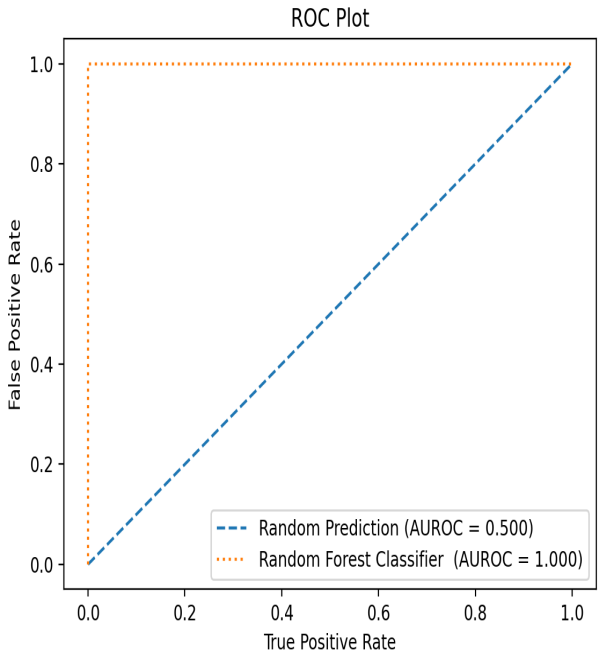

Fig. 6. ROC PLOT

	Precision	Recall	f1-score	Support
0	1.00	0.99	0.99	84
1	0.99	1.00	1.00	160
Accuracy			1.00	244
Macro Avg.	1.00	0.99	1.00	244
Weighted Avg.	1.00	1.00	1.00	244

TABLE XVII
METRICES

	True	False
Positive	83	1
Negative	0	160

TABLE XVIII
CONFUSION MATRIX

G. Support Vector Machines

The objective of the support vector machine [7] algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.

Training Dataset Count	1032
Test Dataset Count	244
Accuracy	0.9631
Random (Chance) Prediction AUROC	0.5
Decision Tree Classifier AUROC	0.9464
Training Time (Sec)	16.57
Testing Time (Sec)	0.16

TABLE XIX
RESULT

	Precision	Recall	f1-score	Support
0	1.00	0.89	0.94	84
1	0.95	1.00	0.97	160
Accuracy			0.96	244
Macro Avg.	0.97	0.95	0.96	244
Weighted Avg.	0.97	0.96	0.96	244

TABLE XX
METRICES

	True	False
Positive	75	9
Negative	0	160

TABLE XXI
CONFUSION MATRIX

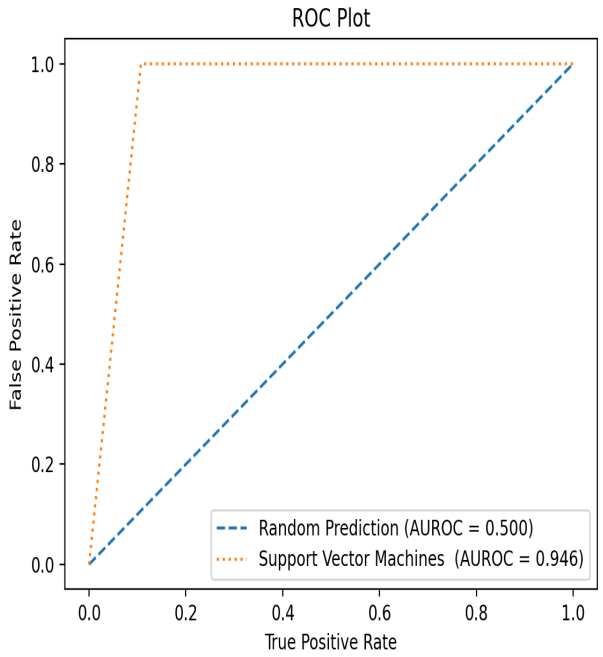

Fig. 7. ROC PLOT

VII. CONCLUSION

	Accuracy	AUROC	Training Time	Testing Time	Score*
Decision Tree Classifier	0.9877	0.9877	7.98	0.16	0.76
Gradient-Boosted Tree Classifier	0.9877	0.9938	15.33	0.16	0.40
Logistic Regression Model	0.9836	0.9982	4.93	0.12	1.66
Multilayer Perceptron Classifier	0.9016	0.9676	16.46	0.15	0.35
Naive Bayes Classifier	0.9754	1.0	5.25	0.18	1.03
Random Forest Classifier	0.9953	1.0	9.29	0.19	0.56
Support Vector Machines	0.9631	0.9464	16.57	0.16	0.34

TABLE XXII
RESULT COMPARISON TABLE

* The score is calculated as (Accuracy * AUROC) / (Training time * Testing Time)

From the above result we can see that the score for the Logistic Regression Model is the highest, hence this means that the accuracy and time to compute the results are at the best possible trade offs for this particular model in our case. Since the model needs to be trained for each player after every session of the game we need a fast and accurate model to provide best results and this model perfectly suits our needs. In production if we need higher accuracies and a lighter performance overhead is acceptable then the Random Forest Classifier will be the best choice to go for, else for resource constrained systems Logistic Regression Model will do the best.

ACKNOWLEDGMENT

This research would not have been possible without the guidance, assistance and suggestions of many individuals. We would like to express our deep sense of gratitude and indebtedness to each and everyone who has helped us. We express our sincere gratitude to Dr. Mohan Babu G.N, Principal, BMSIT & M for providing all the facilities and the support. We heartily thank Dr. Anil G N, Head of Department, Department of Computer Science and Engineering, BMSIT & M for his constant encouragement and inspiration in taking up this topic for research. We gratefully thank our guide, Dr. Anjan Krishnamurthy, Assistant Professor, Department of Computer Science and Engineering, BMSIT & M for encouragement and advice throughout the course of the research. Special thanks to all the staff members of the Computer Science Department and colleagues for their help and kind cooperation.

REFERENCES

[1] Decision trees Accessed on: June. 07, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Decision_tree

[2] Gradient boosted decision trees Accessed on: June. 07, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Gradient_boosting

[3] Logistic regression Accessed on: June. 07, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Logistic_regression

[4] Multilayer perceptron Accessed on: June. 07, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Multilayer_perceptron

[5] Naive Bayes classifier Accessed on: June. 07, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Naive_Bayes_classifier

[6] Random Forest Classification Accessed on: June. 07, 2020. [Online]. Available: https://towardsdatascience.com/random-forest-classification-and-its-implementation-d5d840dbead0

[7] Support Vector Machine — Introduction to Machine Learning Algorithms Accessed on: June. 07, 2020. [Online]. Available: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47