

# **Cervical Cancer Behavior Risk Prediction Using Machine Learning**

by

**Bushra Tabassum (170021010)**

**Nafis Jabid Hasan (170021050)**

**Md. Muhibul Azim (170021055)**

A Thesis Submitted to the Academic Faculty in Partial Fulfillment of the  
Requirements for the Degree of

**BACHELOR OF SCIENCE IN ELECTRICAL AND ELECTRONIC  
ENGINEERING**




Department of Electrical and Electronic Engineering  
**Islamic University of Technology (IUT)**  
Gazipur, Bangladesh

May 2022

## Certificate of Approval

The thesis titled “Cervical Cancer Behavior Risk Prediction Using Machine Learning” submitted by Bushra Tabassum (170021010), Nafis Jabid Hasan (170021050) and Md. Muhibul Azim (170021055) has been found as satisfactory and accepted as partial fulfillment of the requirement for the degree of Bachelor of Science in Electrical and Electronic Engineering on May, 2022.

Approved by:

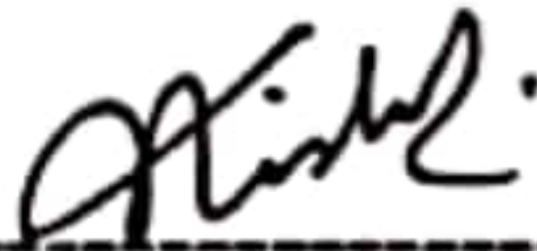


(Signature of the Supervisor)

**Fahim Faisal**

Assistant Professor

Department of Electrical and Electronic Engineering (EEE)  
Islamic University of Technology (IUT)



(Signature of the Co-Supervisor)

**Mirza Muntasir Nishat**

Assistant Professor

Department of Electrical and Electronic Engineering (EEE)  
Islamic University of Technology (IUT)

## Declaration of Authorship

This is to certify that the work presented in this thesis is the outcome of the analysis and experiments carried out by Bushra Tabassum, Nafis Jabid Hasan, and Md. Muhibul Azim under the supervision of Mr. Fahim Faisal, Assistant Professor, Department of Electrical and Electronic Engineering (EEE), Islamic University of Technology (IUT), Gazipur, Dhaka, Bangladesh. It is also declared that neither this thesis nor any part of it has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others have been acknowledged in the text and a list of references is given.

### Authors:

Bushra Tabassum

**Bushra Tabassum**

**Student ID: 170021010**

Nafis Jabid Hasan.

**Nafis Jabid Hasan**

**Student ID: 170021050**

Md Muhibul Azim

**Md. Muhibul Azim**

**Student ID: 170021055**

## ***Dedicated to***

*Our beloved parents & teachers whose support made it all possible for us*

# **TABLE OF CONTENTS**

<b>List of Tables</b> .....	<b>vii</b>
<b>List of Figures</b> .....	<b>ix</b>
<b>List of Acronyms</b> .....	<b>x</b>
<b>Acknowledgements</b> .....	<b>xi</b>
<b>Abstract</b> .....	<b>xii</b>
<b>1. Introduction</b> .....	<b>01</b>
1.1 Introduction .....	01
1.2 Significance of Research .....	02
1.3 Objectives of this Research .....	05
1.4 Main Contribution .....	05
1.5 Thesis Outline .....	06
<b>2. Literature Review</b> .....	<b>07</b>
2.1 Relevant Research .....	07
2.2 Comparative Analysis of Relevant Research .....	09
<b>3. Methodology</b> .....	<b>13</b>
3.1 Basic Methodology .....	13
3.2 Description of the Basic Methodology .....	14
3.3 Description of Performance Matrices in ML .....	14
3.4 Formula of Performance Matrices .....	15
3.5 Detailed Methodology of our Research .....	16

<b>4. Data Preprocessing .....</b>	<b>18</b>
4.1 Importance of Feature Engineering .....	18
4.2 Dataset Description .....	18
4.3 Plotting Correlation Heatmap .....	20
4.4 Dataset Splitting .....	21
<b>5. Introduction to Algorithms .....</b>	<b>22</b>
5.1 K-Nearest Neighbor .....	22
5.2 Random Forest Classifier .....	25
5.3 Support Vector Machine .....	26
5.4 AdaBoost .....	29
5.5 XGBoost .....	31
5.6 Decision Tree Classifier .....	33
5.7 Gaussian Naïve Bayes .....	36
5.8 CatBoost .....	38
5.9 Multi-Layer Perceptron .....	43
5.10 Gradient Boosting Classifier .....	46
5.11 XG Boost with Random Forest .....	49
<b>6. Result and Analysis .....</b>	<b>51</b>
6.1 Implementation of Machine Learning Algorithms .....	51
6.1.1 K-Nearest Neighbors .....	51
6.1.2 Random Forest Classifier .....	52
6.1.3 Support Vector Machine .....	52
6.1.4 AdaBoost .....	53
6.1.5 XGBoost .....	53
6.1.6 Decision Tree Classifier .....	54
6.1.7 Gaussian Naïve Bayes .....	55
6.1.8 CatBoost .....	55
6.1.9 Multi-Layer Perceptron .....	56

6.1.10 Gradient Boosting Classifier .....	57
6.1.11 XG Boost with Random Forest .....	57
6.2 Overall Performance of All the Classifiers .....	58
6.3 Analysis .....	60
<b>7. Conclusion and Future Scope .....</b>	<b>63</b>
7.1 Conclusion .....	63
7.2 Future Scopes .....	63
<b>References .....</b>	<b>64</b>

## LIST OF TABLES

<b>Table 2.1</b>	Comparative Tabulation of Previous Relevant Research .....	09
<b>Table 3.1</b>	Confusion Matrices in Machine Learning .....	14
<b>Table 4.1</b>	Importance of Feature Engineering .....	19
<b>Table 5.1</b>	Performance of Transformation in CatBoost Algorithm .....	39
<b>Table 6.1</b>	Confusion Matrix (Without Hyperparameter Tuning) (KNN) .....	51
<b>Table 6.2</b>	Performance Matrix (Without Hyperparameter Tuning) (KNN) ....	51
<b>Table 6.3</b>	Confusion Matrix (With Hyperparameter Tuning) (KNN) .....	51
<b>Table 6.4</b>	Performance Matrix (With Hyperparameter Tuning) (KNN) .....	51
<b>Table 6.5</b>	Confusion Matrix (Without Hyperparameter Tuning) (RFC) .....	52
<b>Table 6.6</b>	Performance Matrix (Without Hyperparameter Tuning) (RFC) ....	52
<b>Table 6.7</b>	Confusion Matrix (With Hyperparameter Tuning) (RFC) .....	52
<b>Table 6.8</b>	Performance Matrix (With Hyperparameter Tuning) (RFC) .....	52
<b>Table 6.9</b>	Confusion Matrix (Without Hyperparameter Tuning) (SVM) .....	52
<b>Table 6.10</b>	Performance Matrix (Without Hyperparameter Tuning) (SVM) ....	52
<b>Table 6.11</b>	Confusion Matrix (With Hyperparameter Tuning) (SVM) .....	52
<b>Table 6.12</b>	Performance Matrix (With Hyperparameter Tuning) (SVM) .....	53
<b>Table 6.13</b>	Confusion Matrix (Without Hyperparameter Tuning) (AdaB) .....	53
<b>Table 6.14</b>	Performance Matrix (Without Hyperparameter Tuning) (AdaB) ...	53
<b>Table 6.15</b>	Confusion Matrix (With Hyperparameter Tuning) (AdaB) .....	53
<b>Table 6.16</b>	Performance Matrix (With Hyperparameter Tuning) (AdaB) .....	53
<b>Table 6.17</b>	Confusion Matrix (Without Hyperparameter Tuning) (XGB) .....	53
<b>Table 6.18</b>	Performance Matrix (Without Hyperparameter Tuning) (XGB) ....	54
<b>Table 6.19</b>	Confusion Matrix (With Hyperparameter Tuning) (XGB) .....	54
<b>Table 6.20</b>	Performance Matrix (With Hyperparameter Tuning) (XGB) .....	54
<b>Table 6.21</b>	Confusion Matrix (Without Hyperparameter Tuning) (DTC) .....	54
<b>Table 6.22</b>	Performance Matrix (Without Hyperparameter Tuning) (DTC) ....	54
<b>Table 6.23</b>	Confusion Matrix (With Hyperparameter Tuning) (DTC) .....	54
<b>Table 6.24</b>	Performance Matrix (With Hyperparameter Tuning) (DTC) .....	54
<b>Table 6.25</b>	Confusion Matrix (Without Hyperparameter Tuning) (GNB) .....	55



<b>Table 6.26</b>	Performance Matrix (Without Hyperparameter Tuning) (GNB) ....	55
<b>Table 6.27</b>	Confusion Matrix (With Hyperparameter Tuning) (GNB) .....	55
<b>Table 6.28</b>	Performance Matrix (With Hyperparameter Tuning) (GNB) .....	55
<b>Table 6.29</b>	Confusion Matrix (Without Hyperparameter Tuning) (CatB) .....	55
<b>Table 6.30</b>	Performance Matrix (Without Hyperparameter Tuning) (CatB) ....	55
<b>Table 6.31</b>	Confusion Matrix (With Hyperparameter Tuning) (CatB) .....	56
<b>Table 6.32</b>	Performance Matrix (With Hyperparameter Tuning) (CatB) .....	56
<b>Table 6.33</b>	Confusion Matrix (Without Hyperparameter Tuning) (MLP) .....	56
<b>Table 6.34</b>	Performance Matrix (Without Hyperparameter Tuning) (MLP) ....	56
<b>Table 6.35</b>	Confusion Matrix (With Hyperparameter Tuning) (MLP) .....	56
<b>Table 6.36</b>	Performance Matrix (With Hyperparameter Tuning) (MLP) .....	56
<b>Table 6.37</b>	Confusion Matrix (Without Hyperparameter Tuning) (GradB) .....	57
<b>Table 6.38</b>	Performance Matrix (Without Hyperparameter Tuning) (GradB) ..	57
<b>Table 6.39</b>	Confusion Matrix (With Hyperparameter Tuning) (GradB) .....	57
<b>Table 6.40</b>	Performance Matrix (With Hyperparameter Tuning) (GradB) .....	57
<b>Table 6.41</b>	Confusion Matrix (Without Hyperparameter Tuning) (XGBRF) ...	57
<b>Table 6.42</b>	Performance Matrix (Without Hyperparameter Tuning) (XGBRF).	57
<b>Table 6.43</b>	Confusion Matrix (With Hyperparameter Tuning) (XGBRF) .....	58
<b>Table 6.44</b>	Performance Matrix (With Hyperparameter Tuning) (XGBRF) ....	58
<b>Table 6.45</b>	Overall Accuracy of Classifiers from Machine Learning ( Without Hyperparameter Tuning) .....	58
<b>Table 6.46</b>	Overall Accuracy of Classifiers from Machine Learning ( With Hyperparameter Tuning) .....	59

## LIST OF FIGURES

<b>Figure 1.1</b>	FTIR spectrum of healthy cervical cells .....	04
<b>Figure 1.2</b>	FTIR spectra of both healthy and cancerous cells .....	04
<b>Figure 3.1</b>	Basic Methodology Flowchart .....	13
<b>Figure 3.2</b>	Detailed Working Flow of our Research .....	16
<b>Figure 4.1</b>	Correlation heatmap of attributes .....	20
<b>Figure 5.1</b>	KNN Architecture .....	23
<b>Figure 5.2</b>	Random Forest Classifier Architecture .....	25
<b>Figure 5.3</b>	Support Vector Machine Architecture .....	26
<b>Figure 5.4</b>	Linear & Non-linear SVM architecture .....	27
<b>Figure 5.5</b>	Operating Mechanism of Adaboost .....	29
<b>Figure 5.6</b>	AdaBoost Architecture .....	30
<b>Figure 5.7</b>	XGBoost Architecture .....	32
<b>Figure 5.8</b>	Decision Tree Algorithm structure .....	34
<b>Figure 5.9</b>	Class Distribution of GNB in Graphical Representation .....	37
<b>Figure 5.10</b>	Box and whisker plot of XGBRF Ensemble Size vs. Accuracy Assessment .....	50
<b>Figure 5.11</b>	Box and whisker plot of XGBRF Feature Set Size vs. Accuracy Assessment .....	50
<b>Figure 6.1</b>	ROC Curve (without Hyperparameter Tuning) .....	59
<b>Figure 6.2</b>	ROC Curve (with Hyperparameter Tuning) .....	60
<b>Figure 6.3</b>	Grid Search Algorithm Flow Chart .....	61

## **LIST OF ACRONYMS**

<b>WHO</b>	World Health Organization
<b>HPV</b>	Human Papilloma Virus
<b>CART</b>	Classification And Regression Tree
<b>FTIR</b>	Fourier Transform Infrared Spectroscopy
<b>DTC</b>	Decision Tree Classifier
<b>GNB</b>	Gaussian Naïve Bayes
<b>RFC</b>	Random Forest Classifier
<b>KNN</b>	K-Nearest Neighbors
<b>SVM</b>	Support Vector Machine
<b>CatB</b>	CatBoost
<b>MLP</b>	Multi-Layer Perceptron
<b>GradB</b>	Gradient Boosting Classifier
<b>AdaB</b>	AdaBoost
<b>XGB</b>	XG Boost
<b>XGBRF</b>	XG Boost with Random Forest
<b>MRI</b>	Magnetic Resonance Imaging
<b>DWI</b>	Diffusion Weighted Imaging
<b>ROC</b>	Receiver Operation Characteristic
<b>CNN</b>	Convolutional Neural Network
<b>ANN</b>	Artificial Neural Network
<b>GSCV</b>	Grid Search Cross-Validation
<b>SMOTE</b>	Synthetic Minority Over-sampling Technique
<b>DBSCAN</b>	Density-Based Spatial Clustering of Applications with Noise
<b>iForest</b>	Isolation Forest

## **ACKNOWLEDGEMENTS**

Foremost, we would like to express our sincere gratitude and gratefulness to the Almighty Allah; without His graces and blessings, this study would not have been possible. Acknowledging all who helped us complete this work, we wish to compliment the university's significant role and the department that has been very amiable to us during the entire period of our research.

We are indebted to our honorable supervisor, Fahim Faisal sir and honorable co-supervisor, Mirza Muntasir Nishat sir for their selfless support, motivation, patience, enthusiasm, and extensive knowledge of the relevant fields. Their continuous guidance and careful supervision kept us going even during the hardest of hours.

Lastly, our warmest tribute to our parents, family members, and friends, whose moral support and well wishes benefitted us spiritually in achieving our goals.

## **ABSTRACT**

Cervical cancer is a serious public health concern that affects women all over the world. Early risk prediction of cervical cancer can play an essential role in prevention by boosting public awareness of this disease, because it is a fatal disease. Both healthcare professionals and persons at risk can benefit from early prediction utilizing a Machine Learning (ML) model. Using a dataset from the UCI ML repository, eleven supervised machine learning algorithms are used to predict early risks of cervical cancer in this work. Accuracy, precision, F1, recall, and ROC\_AUC are among the performance metrics used to predict the early risks of cervical cancer using machine learning models. Finally, a comparison analysis reveals that using the Multi-Layer Perceptron (MLP) method with default hyperparameters, this study achieved 93.33% prediction accuracy. Decision Tree Classifier (DTC), Random Forest Classifier (RFC), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Multi-Layer Perceptron (MLP) all showed accuracy of 93.33% when using the hyperparameter tuning strategy.

# **Chapter 1**

## **INTRODUCTION**

### ***1.1 Introduction***

Cervical cancer refers to a malignancy that affects a woman's cervix which is considered as one of the most important health issues that affects millions of women worldwide, particularly in developing countries. In 2012, about 271 thousand women died as a result of cervical cancer. According to the World Health Organization (WHO), over 89% of deaths due to cervical cancer occur in developing or underdeveloped nations [1]. If immediate move was not taken, the mortality rate was expected to grow by nearly 24.99% [2]. In 2012, about 445,000 cases were discovered and almost 83% of all were new cases [3].

Symptoms of cervical cancer include irregular periods, unexpected blood, and atypical menstruation. Hence, a pap smear test can diagnose cervical cancer and has been shown to reduce death risk by almost 90% and cervical cancer risk by 60% to 90% [4]. A pap test is a simple, painless, and quick procedure to check for cervical cancer. During a pelvic exam, cells from a woman's cervix are gathered for examination. However, the absence of medical equipment, inadequate nurturing, simple diagnostic reproducibility, careless maintenance, and apathy on the part of the specialists delivering the exam owing to its monotonous behavior are all drawbacks of this examination [5]. Although the pap smear test and HPV vaccine have lowered the number of women under the age of 18 who die from cervical cancer, the death rate in developing nations keeps rising. [6].

The cervix is the bottom, smallest part of the uterus, which is called womb. Moreover, The Human Papillomavirus (HPV) is a cancer-causing virus that can be spread through poor lifestyle choices. Numerous people have been afflicted with the HPV at some point in their lives, and the virus may clear on its own, proving to be harmless. High-risk HPV infection, on the other hand, appears on the cervical surface as flat warts or condylomas that release infective virus particles [7]. As per statistics, half of all cervical cancer cases in the United States occur in women who have never had their cervical cancer checked, with the remaining 10% occurring in women who have not been tested in the previous 5 years [8].

Many attempts have been made to use machine learning to aid in the initial detection of cervical cancer by assisting with tasks such as categorisation. Many machine learning approaches are commonly used, including SVM, KNN, CART, ANN, RFT, and others. We look at the many machine learning models that have been widely used in research in recent years, focusing on the ML algorithms used, datasets used, and early approaches used, applications used, and precision obtained. The Survey's results are reported in the concluding section. The purpose of this study is to gain a full understanding of current work on cervical cancer prognosis using machine learning, as well as to establish the framework for future research in this field.

In this regard, early identification of cervical cancer using lifestyle information can help save many lives. So, collecting sufficient data, analyzing them and finding the hidden pattern can impose a significant contribution in this regard. With the advancement of data science, machine learning techniques are proven to be handy in performing such operations so that prompt detection and timely treatment can be ensured by the healthcare professionals [9][10][11].

## ***1.2 Significance of Research***

Cervical cancer is the second most common cancer in women worldwide, first one is breast cancer [12]. It is also the major cause of death when compared to other types of cancer. In the United States, however, the figures are quite different.

Cervical malignancy is the eighth most common malignancy in women, with incidence and mortality rates less than half of the global average. This disparity may be due in part to the developed world's success in developing universal and effective diagnostic techniques such as the Pap test. As a result, accurate cervical cancer screening is a critical tool to combat.

- ***Pap smear test***

Cervical cancer is first detected with a Pap test.

Dr. Georgios Papanikolaous [13], an American, created this test in the 1940s. Cell samples are taken from a specific region of the cervix using a simple cervical swab. These cells are then tested for anomalies inside a research lab. Although aberrant outcome do not always mean the patient has cervical cancer, they do indicate that alterations in the cervical cells are occurring and that additional treatment is required.

- ***Colposcopy Procedure***

To further investigate abnormal Pap smear results, a colposcopy and biopsy are usually conducted. This procedure is extremely accurate, and it is sometimes referred to as the "gold standard" cervical cancer test.

This usually implies that it is the most accurate test for diagnosing this condition at the moment. It can theoretically be 100 percent accurate, but no medical diagnostic method is perfect, so this is rarely the case.

Colposcopy is a diagnosing treatment that includes using a colposcope, which is equivalent to a binocular microscope, to examine an enlightened as well as enlarged perspective of the vulva, cervix, and vagina. The colposcopist differentiates normal from diseased cells during this examination and obtains biopsies as needed for additional pathological investigation.

When compared to the colposcopy, The Pap smear is a straightforward, cost-effective procedure. However, it is not disease sensitive and has lower specificity , accuracy, and sensitivity.

Although colposcopy is extremely accurate, it is both time consuming and resource costly. The method also necessitates the use of specially trained staff (colposcopists).

This is used for follow-up recommendations associated with abnormal or unclear Pap smear results because of these concerns. Furthermore, it is obtrusive, potentially unpleasant, and undeniably inconvenient for many women.

As a result, there seems to be a requirement for a quasi-specialist, extremely accurate, and low-cost method to increase Pap smear sensitivity. It would be advantageous to health surveillance professionals generally , but notably in the developing nations, where cervical cancer is the main cause of death among young women due to a lack of testing.



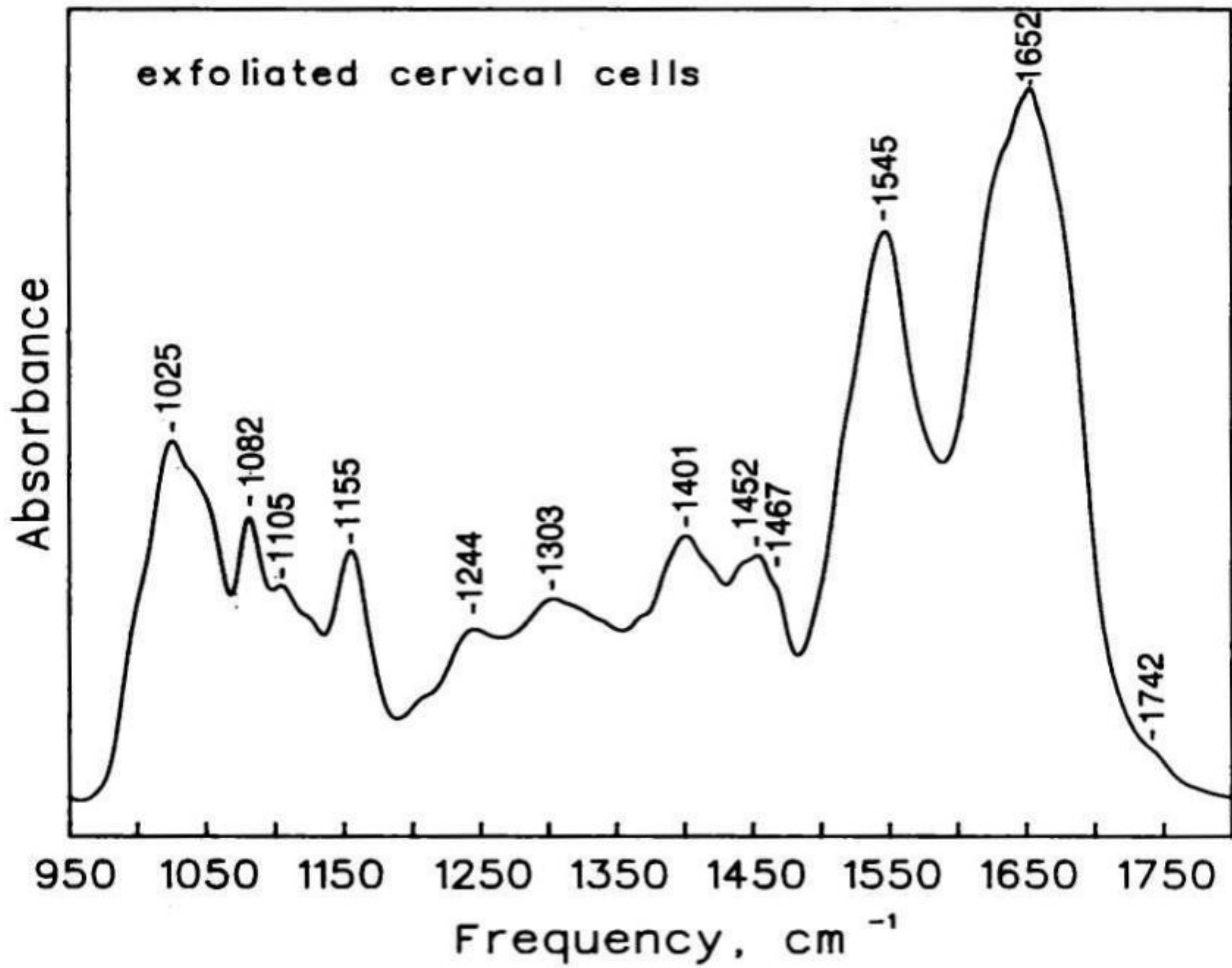


Figure. 1.1 FTIR spectrum of healthy cervical cells [14].

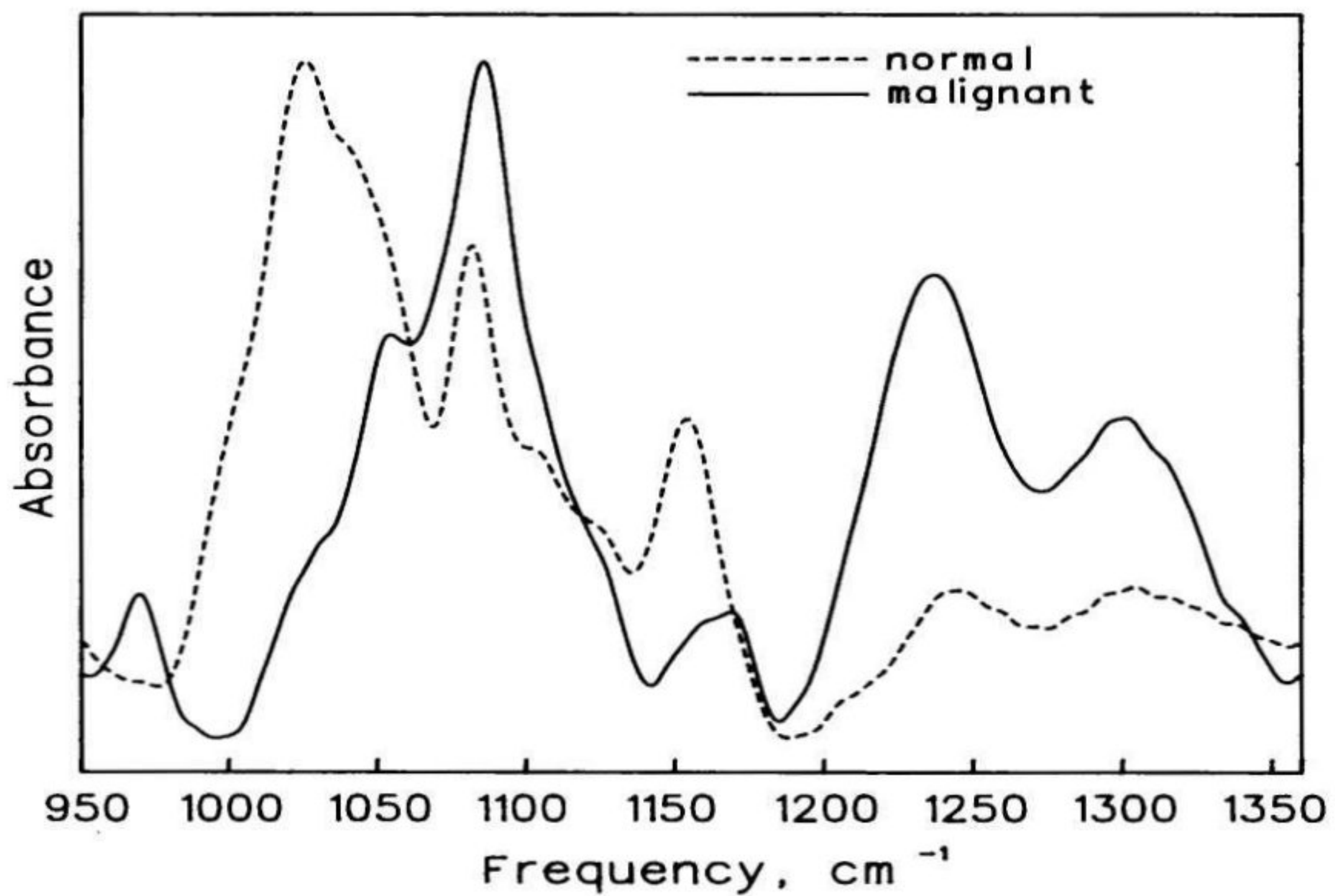


Figure. 1.2. FTIR spectra of both healthy and cancerous cells [14].

### ***1.3 Objectives of this Research***

The main purpose is to forecast cervical cancer behavior risk. Eleven Machine Learning Algorithms are used to accurately predict whether or not a patient has cancer cells. On a dataset relevant to cervical cancer risk prediction in women, we used the Tree Classifier (DTC), Gaussian Nave Bayes (GNB), Random Forest Classifier (RFC), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), CatBoost (CatB), Multi-Layer Perceptron (MLP), Gradient Boosting Classifier (GradB), AdaBoost (AdaB), XG Boost (XGB), XG Boost with Random Forest (XGBRF). We can choose the most suitable approach based on the various results.

Treatment can be started early and lives can be saved if cancer cells can be detected fast using machine learning. Findings of this study could have a noteworthy impact on computer-assisted diagnosis or the development of an e-healthcare system.

Magnetic Resonance Imaging (MRI) and Diffusion-Weighted Imaging (DWI) approaches have been shown to help categorize cervical cancer in the past [15,16]. Cervical cancer a prominent cause of mortality in low-income countries due to a lack of doctor skills and shortage of medical equipment [17].

Cervical cancer incidence varies significantly over the world as a result of preventive measures implemented by affluent countries [18]. Despite advancements in cervical cancer prevention, screening, analysis, and treatment over the last decade, cervical cancer findings differ significantly locally and globally, prompting worldwide gynecological cancer organisations to issue verified management standards to improve patient treatment. [19].

Cervical cancer is now diagnosed mostly through histological and morphological exams that lack well-defined accuracy, specificity ,and sensitivity [20].

### ***1.4 Main Contribution***

We've analyzed the Cervical Cancer dataset with eleven machine learning algorithms (KNN, RFC, SVM, AdaB, XGB, DTC, GNB, MLP, GradB, XGBRF, CatB) We've performed hyperparameter tuning for each of the algorithms in order to select the right set of hyperparameters giving the best accuracy, portrayed Confusion Matrices, calculated individual & overall Performance matrices (Precision, Recall, Accuracy, F-1 score) for finding out the best algorithm among these. Later we've proposed an MLP algorithm that offers the **highest accuracy of 93.33%** among the recent works performed for multiclass classification of this dataset.

## ***1.5 Thesis Outline***

In chapter – 1, the significance of research on Cervical cancer and its data is explained. The necessity of Machine Learning algorithms for increasing the accuracy of diagnostics & the main contribution of our research work is also described.

In chapter – 2, relevant research works published recently on Cervical cancer data classification using ML algorithms are analyzed comparatively that in-line with our research topic of interest.

In chapter – 3, the central architecture & methodology of our entire work, performance matrices that we calculated are narrated.

In chapter – 4, the Data preprocessing & Feature engineering process that we've performed are visually depicted, which is an integral part of any ML research.

In chapter – 5, the 11-Machine learning algorithms & their working procedures are briefly described that we've implemented & optimized to compare the performance matrices of the algorithms.

In chapter – 6, it describes the confusion & performance matrices of all the 11 algorithms & compares the overall result & performance among the algorithms, which portrays the highest accuracy in the classification models using Hyperparameter Tuning.

In chapter – 7, Finally, the discussion of our research work is concluded by mining out the Future-scopes to improve & better implement our study of interest.

## **Chapter 2** **LITERATURE REVIEW**

### ***2.1 Relevant Research***

The ability to make intelligent decisions without the involvement of humans is known as artificial intelligence. This is a new field of computer science focusing on mechanizing greater intelligence functions that do not require human intervention. The ability of a person to acquire and apply knowledge is referred to as intelligence. As a result, intellectual abilities in a computer must be developed through learning from current information in the particular field. Data collection in medical science has grown in popularity since the dawn of the digital revolution. As a result, one of the most serious limitations of machine learning in medical science has been resolved. [21][22][23].

SVM is a type of machine learning used to diagnose cervical cancer and analyze other health data. The SVM has been used to address single and multiclass classification and regression problems, as well as outlier detection. The two basic functions of support vector machines are solving algebraic expressions and forecasting the position of the higher dimensional space. This approach was initially accomplished using well-known quadratic optimization methods. This technique applies to small data sets of around a thousand samples. When dealing with massive amounts of data, a more tailored strategy is required, one that looks at a wider range of factors .

This literature review provides a concise summary of existing studies on ML methods for detecting cervical malignancy. The overview explains the different methods and methodologies that were used. Wen Wu [24] used SVM, SVM-RFE, and SVM-PCA algorithms to examine the data. The data was collected at the Universitario de Caracas Hospital in Caracas and preserved at the University of California, Irvine's repository (UCI). Both responsiveness and expected accuracy were considered during the categorization process. According to the authors, SVM-RFE and SVM-PCA are more effective with limited parameters, though SVM performs well. Although SVM behaves well, the authors believe that SVM-RFE and SVM-PCA perform better with fewer features.

Yulia Ery Kurniawati [25] classified approximately 80 pap smear results as cancerous or non-cancerous. To analyze the data, the authors used three different classification algorithms: SVM, Naive Bayes(NB) , and Random Forest Tree (RFT). The data was shuffled thirty times before characterization using the NBCr, SVM, and RFT, as well as a 10-fold cross-validation

assessment method. The researchers state that the RFT outperforms the other classifiers with an accuracy of around 79%.

To forecast cervical cancer data, Priyanka K Malli [26] employed the KNN approach and ANN. To finish the assignment, the Pap test Graphics dataset was used. The researchers used MATLAB software to make prognostications, and the prognostication systems' accuracy was used to demonstrate their superiority. The researchers also included nearly 19 features and discovered the fact which monitoring the stage of malignancy could help the model even more. To assess if cells were cancerous or benign, R. Vidya [27] employed the Hybrid Induction Technique. The CART, the RFT, and the RFT and K-Means Algorithm were all put to the test. The research was carried out using MATLAB program. The combined effect of RFT and K-means yields the highest accuracy of the three approaches, which is immediately apparent.

Debashree Kashyap [28] used classification methods to divide the cervical malignancy pap smear collection of pictures into five main categories. This method, in addition to image classification, has the additional benefit of assisting in the identification of features for all types of classes using PCA. The authors used a gaussian filter for preprocessing and independent level sets for segmentation. The technique includes a total of 22 qualities. According to the research, the Multi-class Gaussian RBF SVM classifier, Multi-class Polynomial classifier, and Multi-class Quadratic SVM classifier have a maximum accuracy of 95%.

Erick Njoroge [29] described a cervical cancer screening method based on Fourier-Transform Infrared [FTIR] spectroscopic data. The study's goal was to figure out how to improve the commonly used pap test. The authors proposed and compared the FTIR method to previous SVM-based methodologies.

Fazal et al., on the other hand, proposed a cervical cancer prediction model that used DBSCAN (density-based spatial clustering of applications with noise) and isolation forest (iForest) as outlier removers, as well as SMOTE and SMOTE with Tomek link and random forest (RF) classifiers to categorize the data, with a maximum accuracy of 99.5 percent [30].

According to the study, cervical screening using FTIR spectroscopy data has an accuracy rate of 72.99 percent. Jonghwan Hyeon [31] suggested using a pre-trained CNN as a feature extractor to distinguish cells as normal or abnormal to aid in cervical cancer prediction. The authors fed the obtained attributes into four classifiers in order to construct the final forecast. The four classification techniques used were Random Forest, Support Vector Machines, Logistic Regression, and AadaBoost. The procedure was carried out with the assistance of the MATLAB program. In the experiment, SVMs outperformed all other machines. According to the authors, using a more advanced convolutional neural network could improve the model's accuracy.

Kasemsit Teeyapan [32] investigated cervical cancer prediction and tested SVM, THSVM, and TWSVM. 300 samples were provided by Thailand's Lampang Cancer Hospital for the study. The MATLAB software was used to implement two classification methods: binary (two classes) and four-class classification. The Gaussian kernel was chosen due to its superior overall performance. According to the authors, the TWSVM outperformed the others due to its high accuracy and low rate of misclassification of positive data as negative. THSVM outperformed the other two algorithms in one of the binary classification instances.

Sobar employed a machine learning classifier to predict the risk of cervical cancer based on behavior. They discovered a maximum accuracy of 91.68 percent using two common machine learning techniques as classifiers, Nave Bayes (NB) and Logistic Regression (LR) [33].

## 2.2 Comparative Analysis of Relevant Research

**Table 2.1:** Comparative tabulation for a better understanding of previous relevant research

No.	Authors	Year	Title	Summary
1	Debashree Kashyap, Abhishek Somani, Jatin Shekhar, Anupama Bhan and Malay Kishore Dutta	2016	Cervical Cancer Detection And Classification Using Independent Level Sets And Multi SVMs	<p>The Pap smear test has proven to be a reliable method of detecting cervical cancer stages. It debuted in nearly 1941. Examiners struggle to collect and categorize Smear test data in order to detect cervical cancer through manual screening, increasing the risk of human error. The researchers created an autonomous system for detecting and grading the grade of cervical cancer by using geometric and textural aspects of Pap smear images, as well as multi SVM classification.</p> <p>The geometric features used to determine whether a cell is cancerous or normal are created by segmenting the nucleus and cytoplasm with distinct level sets and comparing them to the ground truth. The images are classified with 95% accuracy by extracting well-defined GLCM texture characteristics and combining PCA with the best multi SVM class. In the future, researchers could use a shared database to separate overlapping cells in cervical Pap screening images, extract more features,</p>

				and apply different classifiers. The project could be improved by increasing the contrast enhancement technique to reduce the effect of unwanted background information on Cervical Cell of Pap Smear Images.
2	Erick Njoroge, Stephen R. Alty, Mahbub R. Gani, Maha Alkatib	2006	Classification of Cervical Cancer Cells using FTIR Data	<p>Researchers are looking for non-expert (automated) ways to accurately screen cervical smears for cancer indications due to high false-negative rates of the Papanicolaou smear test and a shortage of colposcopists. The use of Fourier-Transform Infrared (FTIR) spectroscopy has been shown to improve test accuracy. This paper describes the use of Support Vector Machines (SVM) machine learning methodology with FTIR data to augment and improve the standard Pap test.</p> <p>A cohort of 53 participants was used to compare the accuracy of the Pap smear results and the FTIR-based classifier to the findings of the colposcopists. The Pap test had a total classification rate of 43%, while our method had a rate of 72%. This would be quite appealing as a replacement for the Pap smear, and if it is simple to administer, it could be especially valuable in underdeveloped countries where mortality rates from this type of cancer are high. To get closer to the accuracy of the gold standard colposcopy, the authors intend to expand the study to include a larger number of patients and improve the categorization technique.</p>
3	Muhammad Fazal Ijaz, Muhammad Attique, and Youngdoo Son	2020	Data-Driven Cervical Cancer Prediction Model with Outlier Detection and Over-Sampling Methods	The cervical cancer prediction model (CCPM) developed in this study uses risk factors as inputs to enable early cervical cancer prediction. The CCPM begins by removing outliers using outlier detection methods such as density-based spatial clustering of applications with noise (DBSCAN) and isolation forest (iForest), as well as rationally increasing the number of instances in the dataset using SMOTE and SMOTE with Tomek link (SMOTETomek). A random forest (RF) classifier is then used.

				<p>Furthermore, RF outperformed several well-known classification algorithms. Furthermore, the proposed cervical cancer prediction method outperformed previously proposed methods. A smartphone application is also being developed to collect cervical cancer risk factor data and provide CCPM results in the early stages of cervical cancer for immediate and appropriate action.</p> <p>A disadvantage of this study is that it only uses one dataset. In the future, the suggested CCPM could be applied to a variety of cancer datasets to improve the presentation and validity of the results (including liver, breast, prostate, lung, kidney, and thyroid). Another disadvantage is that the method (which combines outlier techniques and data balancing with RF) is slower and uses more memory, but it generates higher accuracy.</p>
4	Marc Arbyn, Elisabete Weiderpass, Laia Bruni, Silvia de Sanjosé, Mona Saraiya, Jacques Ferlay, Freddie Bray	2018	Estimates of incidence and mortality of cervical cancer in 2018: a worldwide analysis	<p>This global study made use of the Global Cancer Observatory 2018 database, which included cancer estimates from 185 countries. We used a hierarchy of methodologies to estimate cervical cancer incidence based on the availability and quality of source data from population-based cancer registries. They used the WHO mortality database to calculate cervical cancer mortality.</p> <p>Based on their Human Development Index, countries were divided into 21 subcontinents and classified as high-resource or low-resource. The number of cases and deaths from cervical cancer in a given country, as well as the directly age-standardised incidence and mortality rate, the indirectly age-standardised incidence and mortality rate, the cumulative incidence and mortality rate, and the average age at diagnosis, were all calculated.</p>



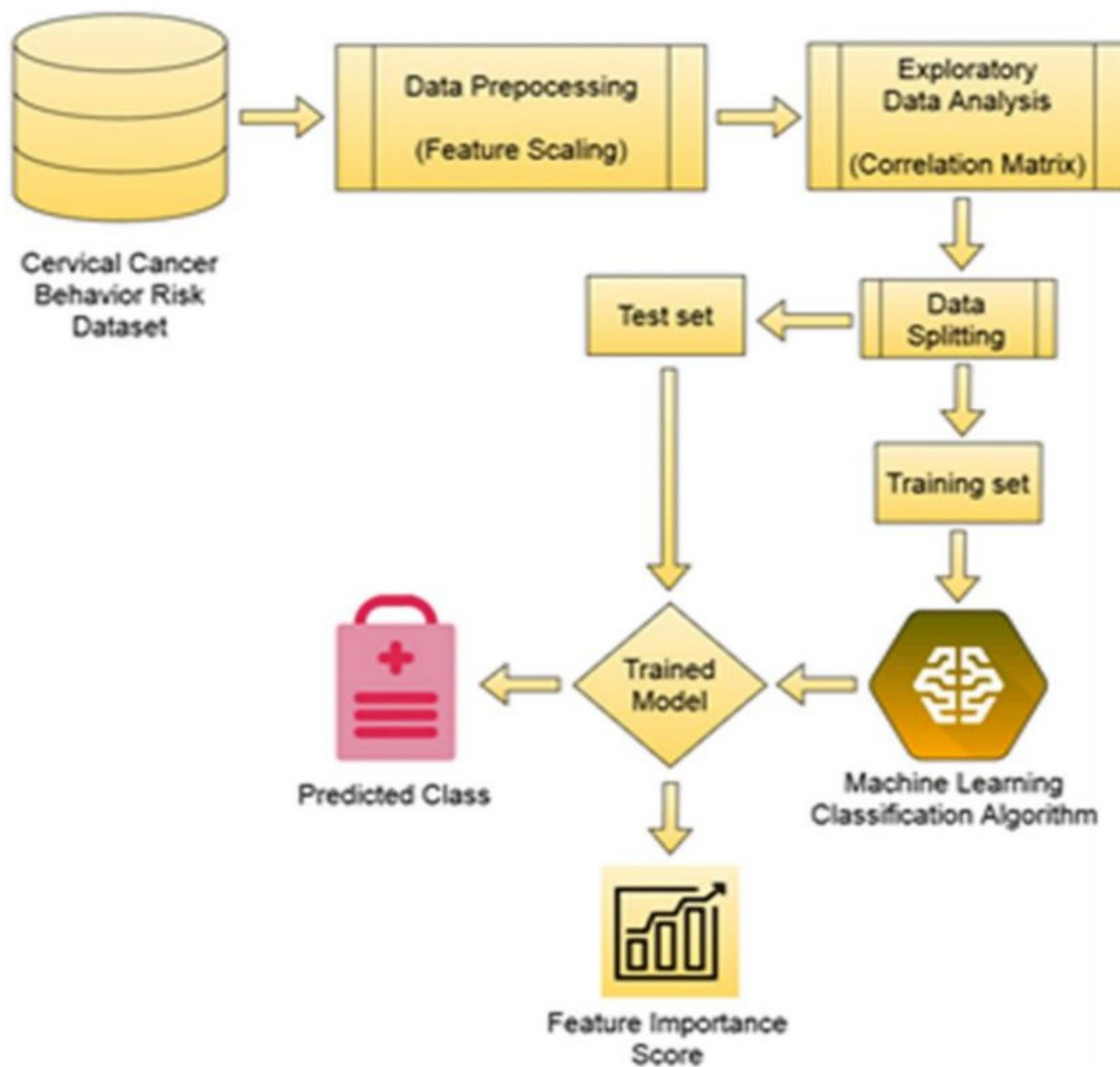
5	Wen Wu, and Hao Zhou	2017	Data-Driven Diagnosis of Cervical Cancer With Support Vector Machine-Based Approaches	<p>Three SVM-based algorithms are used to classify a cervical cancer dataset in this study, and some cervical cancer risk variables are presented. SVM-recursive feature eradication and SVM-principal component analysis (SVM-PCA) are two improved SVM techniques that can be used to diagnose malignant cancer samples.</p> <p>Four target variables represent cervical cancer data: Hinselmann, Schiller, Cytology, and Biopsy. The three SVM-based approaches successfully diagnosed and classified all four targets. After that, a comparison of these three methodologies is performed. It has been demonstrated that the SVM-PCA technique outperforms the others.</p>

# Chapter 3

## METHODOLOGY

### 3.1 Basic Methodology

The basic working flow of our research is depicted by a flowchart below:



**Figure 3.1: Basic Methodology Flowchart**

### ***3.2 Description of the Basic Methodology***

This dataset was obtained from the UCI machine learning repository and it contains information about cervical cancer risk behavior [34]. The dataset consists of 72 instances and 19 attributes, including one target column. There are no missing values in the dataset, and all of the attributes are numerical type.

Exploratory data analysis is carried out after obtaining the Cervical cancer Behavior risk dataset from the UCI repository. The data is then divided into two groups: training and testing. Hyperparameter tuning is applied to the trained data. The ML classifier is then used to evaluate and analyze performance. This is how model selection is carried out. Respective confusion matrices were portrayed. Finally, the performance of each model was analyzed using various performance matrices like accuracy, precision, recall & F-1 score.

### ***3.3 Description of Performance Matrices in ML***

Performance Matrices of ML come from the confusion matrices. Confusion matrices give the visualization of the performance of the ML models. Each row of the matrix presents the instances of the actual class, where each column represents the instances of the predicted class or vice-versa [35]. One example of the confusion matrix is mentioned below:

**Table 3.1:** Confusion Matrices in ML

<b>Confusion Matrix</b>		<b>Predicted</b>	
		<b>True</b>	<b>False</b>
<b>Actual</b>	<b>True</b>	True Positive (TP)	False Negative (FN)
	<b>False</b>	False Positive (FP)	True Negative (TN)

Here, we can see four terms like True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). Their description is given below:

**a) True Positive (TP):** A class was predicted Positive, which is True actually.

**b) True Negative (TN):** A class was predicted Negative, which is True actually.

**c) False Positive (FP):** A class was predicted Positive, which is False actually.

**d) False Negative (FN):** A class was predicted Negative, which is False actually.

### ***3.4 Formula of Performance Matrices***

Formulae that were used to measure the performance matrices of ML and DL models in our research are mentioned below [36]:

$$\mathbf{a)} \text{ Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

$$\mathbf{b)} \text{ Precision} = \frac{TP}{TP + FP}$$

$$\mathbf{c)} \text{ Recall} = \frac{TP}{TP + FN}$$

$$\mathbf{d)} \text{ F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

### 3.5 Detailed Methodology of our Research

The detailed working flow of our research is illustrated below:

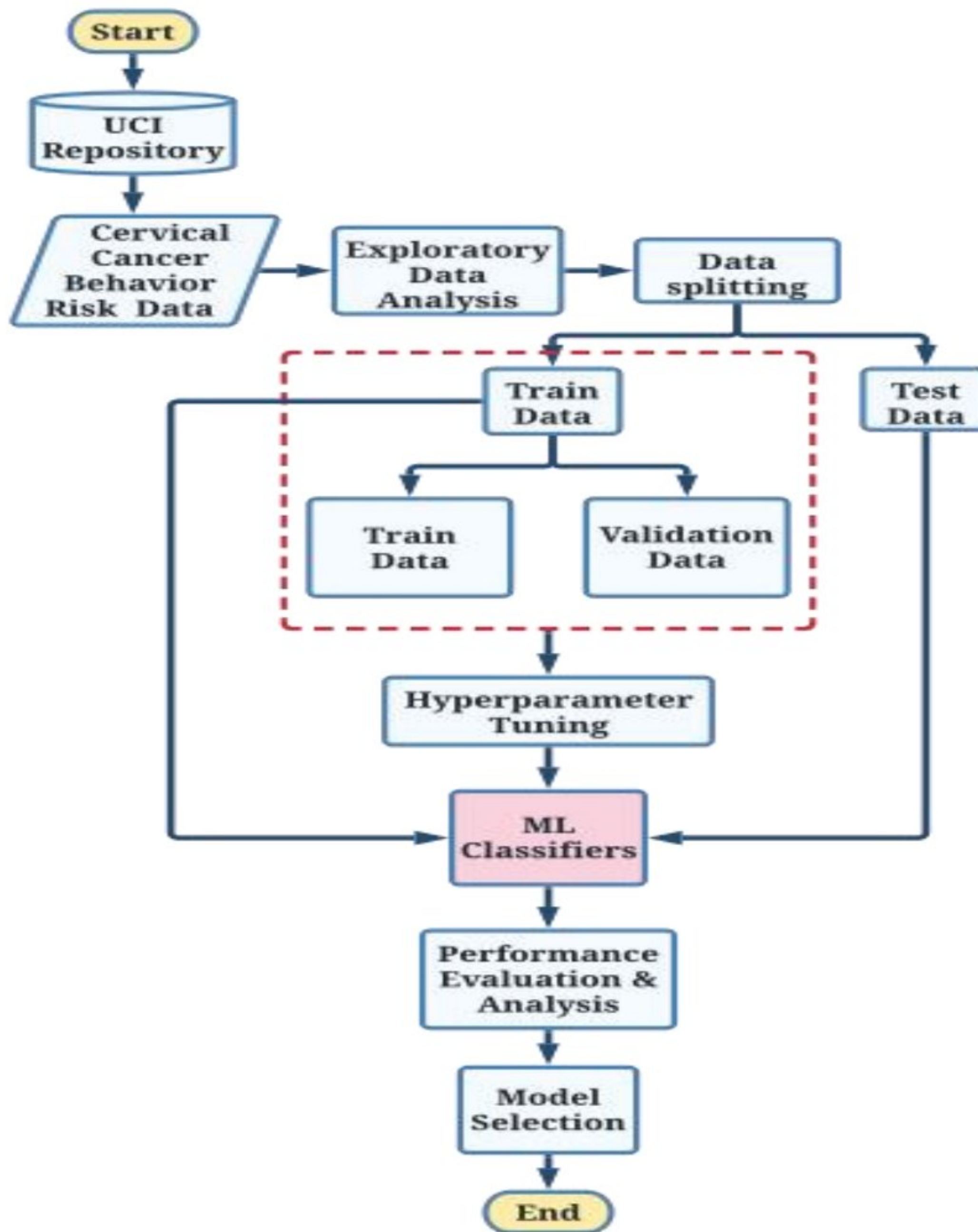


Figure 3.2: Detailed Working Flow of our Research

The dataset was divided into 80:20 ratios for training and testing ML algorithms upon exploratory data analysis. This study used two approaches: one used default hyperparameters of ML algorithms, while the other one used Grid Search Cross-Validation with a 10-fold method to tune hyperparameters. Eleven supervised machine learning algorithms, such as Decision Tree Classifier (DTC), Gaussian Naïve Bayes (GNB), Random Forest Classifier (RFC), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), CatBoost (CatB), Multi-Layer Perceptron (MLP), Gradient Boosting Classifier (GradB), AdaBoost (AdaB), XG Boost (XGB), XG Boost with Random Forest (XGBRF) are trained using train dataset. Then the performance metrics such as accuracy, precision, F1 score, recall, and ROC\_AUC are evaluated for both the default hyperparameter method and the tuned hyperparameter approach using the test dataset.

## **Chapter 4**

### **DATA PREPROCESSING**

#### ***4.1 Importance of Feature Engineering***

The most informative property or attribute related to any dataset to classify different dataset patterns is known as features. Feature Engineering is an integral part of any machine learning and deep learning model. A good feature engineering technique can increase the efficiency & overall performance of the model effectively [37].

The raw data collected from patient history & diagnostics are processed to transform them into useful features that are compatible with algorithm requirements & better represent the underlying & root cause of the disease to the predictive machine learning & deep learning models. The procedure is known as Feature Engineering. Feature engineering converts the data inputs into the most valuable assets that the classifier algorithm can understand & work on.

Different feature engineering techniques are [38]:

- a) Data visualization
- b) Finding missing values & Data imputation
- c) Outlier mining & Handling outliers
- d) Histogram Plotting & Data Binning
- e) Data resampling
- f) Plotting Correlation Heatmap
- g) Dataset splitting

#### ***4.2 Dataset Description***

This dataset was obtained from the UCI machine learning repository and it contains information about cervical cancer risk behavior [34]. The dataset consists of 72 instances and 19 attributes, including one target column. There are no missing values in the dataset, and all of the attributes are numerical type.

**Table 4.1 : Statistical Information of Numeric Attribute**

<b>Sl. No.</b>	<b>Numeric Attributes</b>	<b>Max-Min</b>	<b>Mean</b>	<b>Standard Deviation</b>
1	behavior_sexualRisk	10-2	9.67	1.19
2	behavior_eating	15-3	12.79	2.36
3	behavior_personalHygine	15-3	11.08	3.03
4	intention_aggregation	10-2	7.90	2.74
5	attitude_consistency	15-6	13.35	2.37
6	intention_commitment	10-2	7.18	1.52
7	attitude_spontaneity	10-4	8.61	1.52
8	norm_significantPerson	5-1	3.13	1.85
9	norm_fulfillment	15-3	8.49	4.91
10	perception_vulnerability	15-3	8.51	4.28
11	perception_severity	10-2	5.39	3.40
12	motivation_strength	15-3	12.65	3.21
13	motivation_willingness	15-3	9.69	4.13
14	socialSupport_emotionality	15-3	8.09	4.24
15	SocialSupport_appreciation	10-2	6.16	2.90
16	socialSupport_instrumental	15-3	10.38	4.32
17	empowerment_knowledge	15-3	10.54	4.37
18	empowerment_abilities	15-3	9.32	4.18
19	empowerment_desires	15-3	10.28	4.48
20	ca_cervix	1-0	0.29	0.46

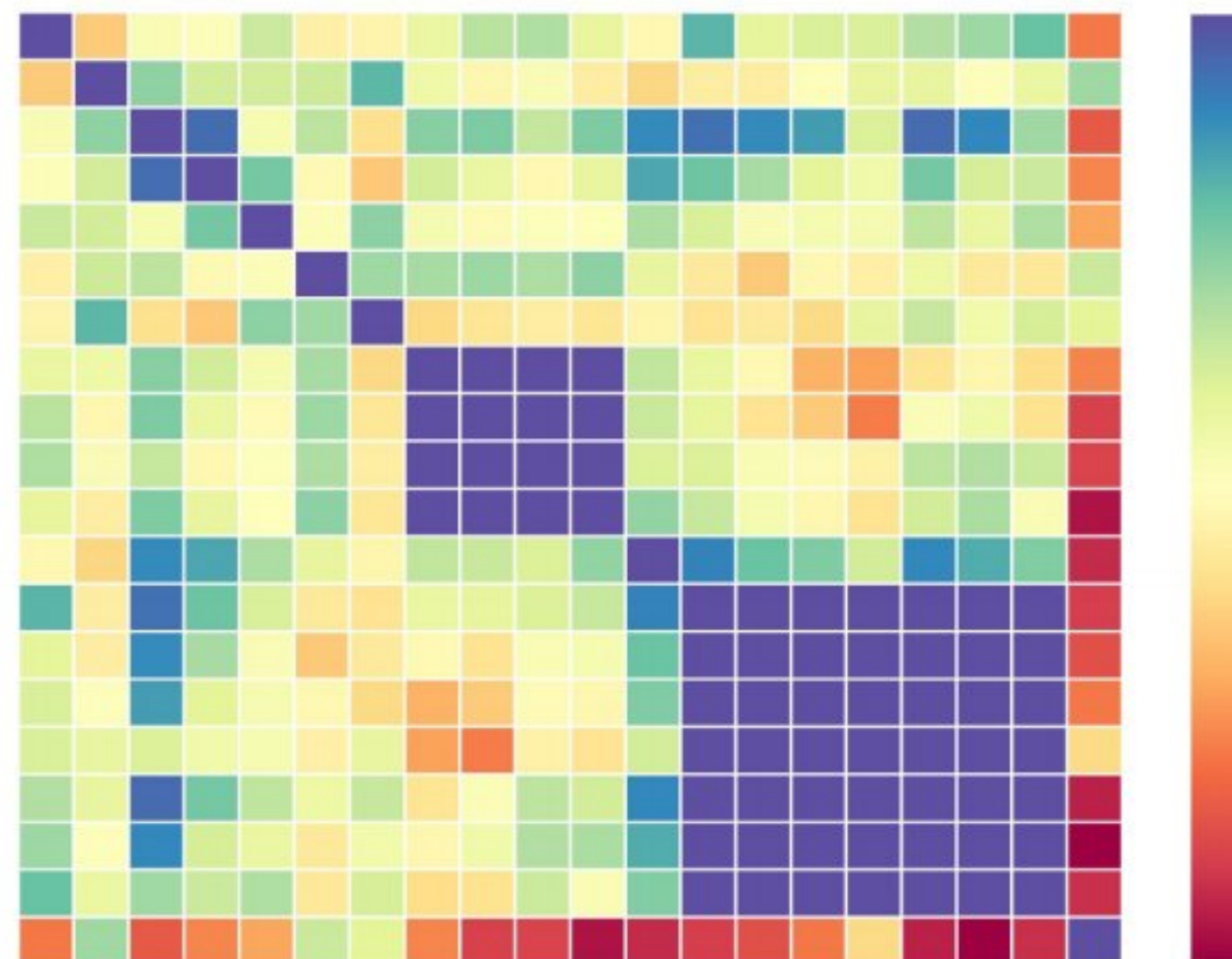


### 4.3 Plotting Correlation Heatmap:

To portray the same dataset in a visually appealing way, it's an essential step in exploratory data analysis, which graphically shows correlating variables, degree & direction of correlation and notify us about multicollinearity problems.

The degree of correlation between variables varies from -1 to +1. Correlating values near to -1 defines the correlation between two variables as more negative (as one value increases, the other value decreases); the closer to -1, the stronger the relationship is. Correlating values near 0 define no linear trend between the two variables. Correlating values near 1 represent the positive linear trend between the two variables.

A correlation heatmap was plotted to view highly correlated dataset features with the target output variable.



**Fig.4.1 Correlation heatmap of attributes**

Exploratory data analysis was performed on the dataset, and a correlation heatmap was created and portrayed in Fig. 4.1, defining the correlation between attributes.

#### ***4.4 Dataset Splitting:***

Data were randomly shuffled & then split into 80:20 ratio for training and testing, respectively, as this ratio shows the least bias and variance for the machine learning and deep learning algorithms we implemented.

This study used two approaches: one used default hyperparameters of ML algorithms, while the other one used Grid Search Cross-Validation with a 10-fold method to tune hyperparameters.

## *Chapter 5* *INTRODUCTION TO ALGORITHMS*

### ***5.1 K-Nearest Neighbors [39,40]***

K-nearest neighbors (KNN) is used for both regression & classification problems, but it's

termed as typically simple, nonparametric & lazy supervised algorithm.

It has the simplest working procedure based on distance measurement:

Step 1 – At first, the training & test dataset is being loaded.

Step 2 – Then nearest 'k' (any integer) number of data points are chosen.

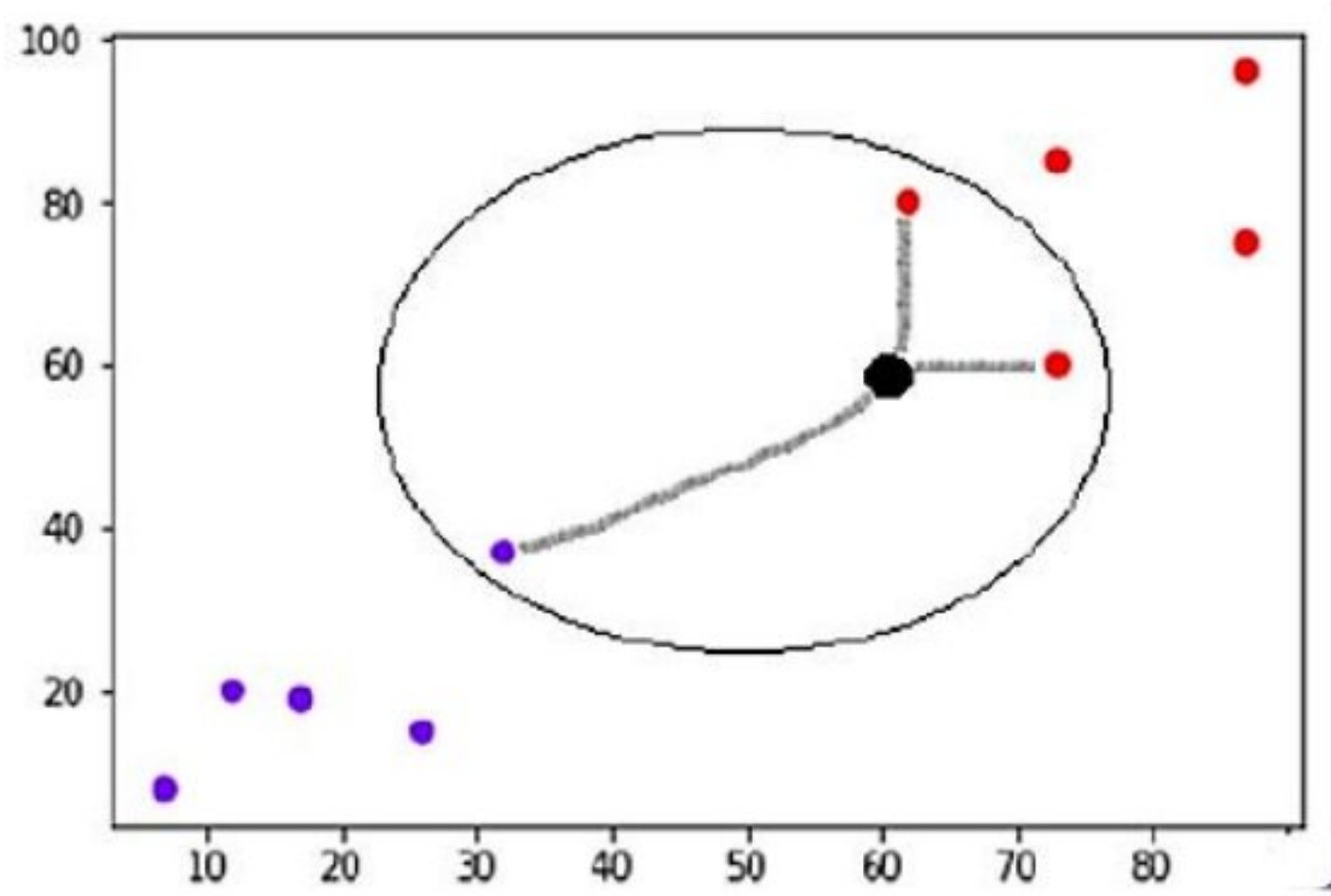
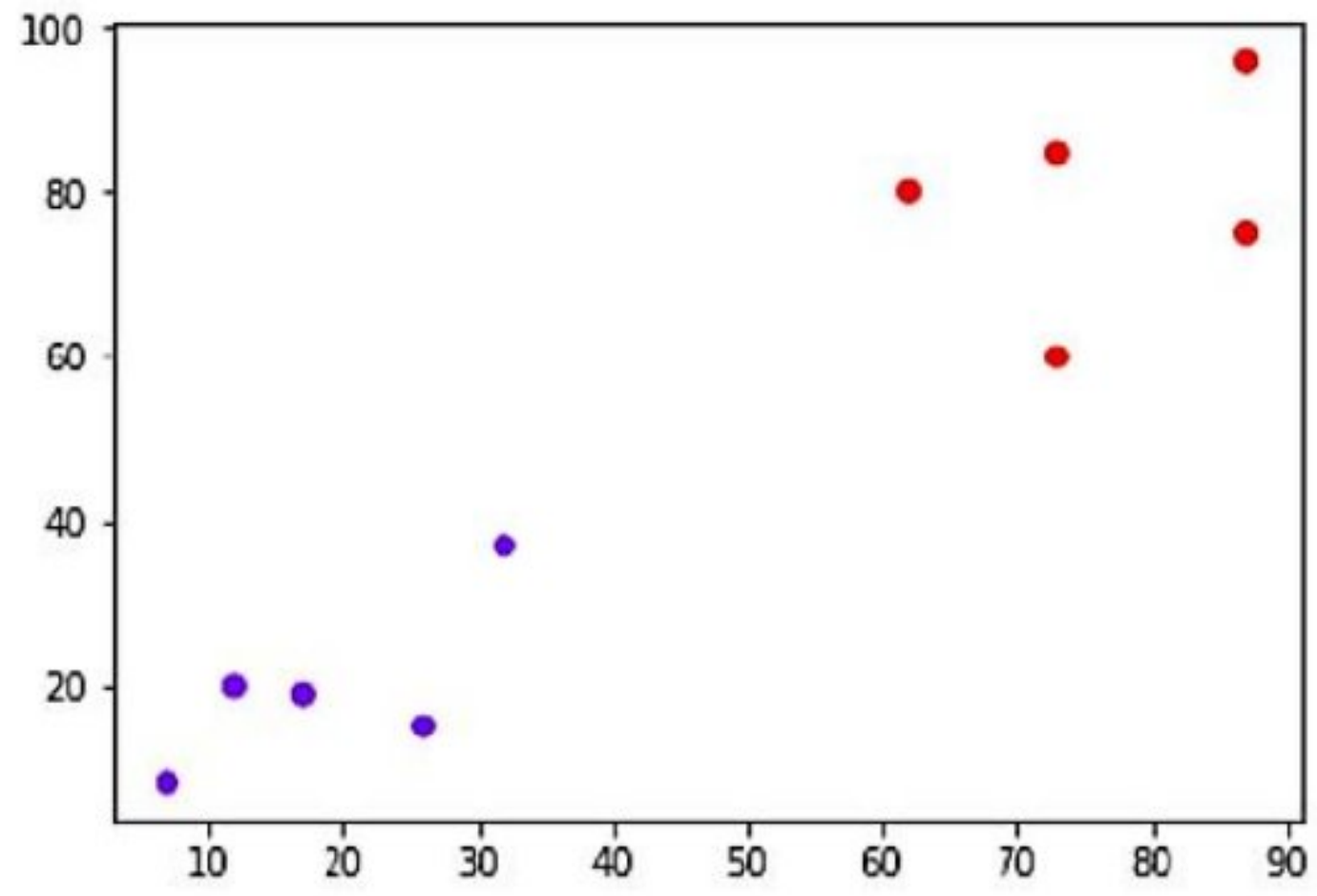
Step 3 – Next, the following procedures are performed on every data point in the test dataset –

a – Distance between each row of training data point and test data point is being measured.

b – The test data points are sorted in ascending order based on the measured distance.

c – Then, from the sorted array, upper K rows are being chosen in order to assign a class to the test data point depending on the most vibrant class of those sorted rows.

Step 4 – End of the algorithm



**Figure 5.1: KNN Architecture [40]**

Distance functions:

i. Euclidean distance:

$$\sqrt{\sum_{j=1}^k (p_j - q_j)^2}$$

ii. Manhattan distance:

$$\sum_{j=1}^k |p_j - q_j|$$

iii. Minkowski distance:

$$\left( \sum_{j=1}^k (|p_j - q_j|)^n \right)^{\frac{1}{n}}$$

iv. Hamming distance:

$$D_H = \sum_{j=1}^k |p_j - q_j|$$

$$p = q \Rightarrow D = 0$$

$$p \neq q \Rightarrow D = 1$$

e.g.:

p	q	Distance
Vanilla	Vanilla	0
Vanilla	Chocolate	1

## 5.2 Random Forest Classifier [41]

**Introduction:** Branched-trees are the indistinguishable property of a forest; the more trees, the more robust the forest is. Random Forest algorithm is another type of supervised learning technique applied in the tasks where Regression or Classification of data are required. Random Forest classifier contains a couple of decision trees composed of different sub-groups of the input dataset, and output from each tree is taken average in order to boost the predictive precision of that dataset. A greater number of decision trees increases accuracy & prevents overfitting.

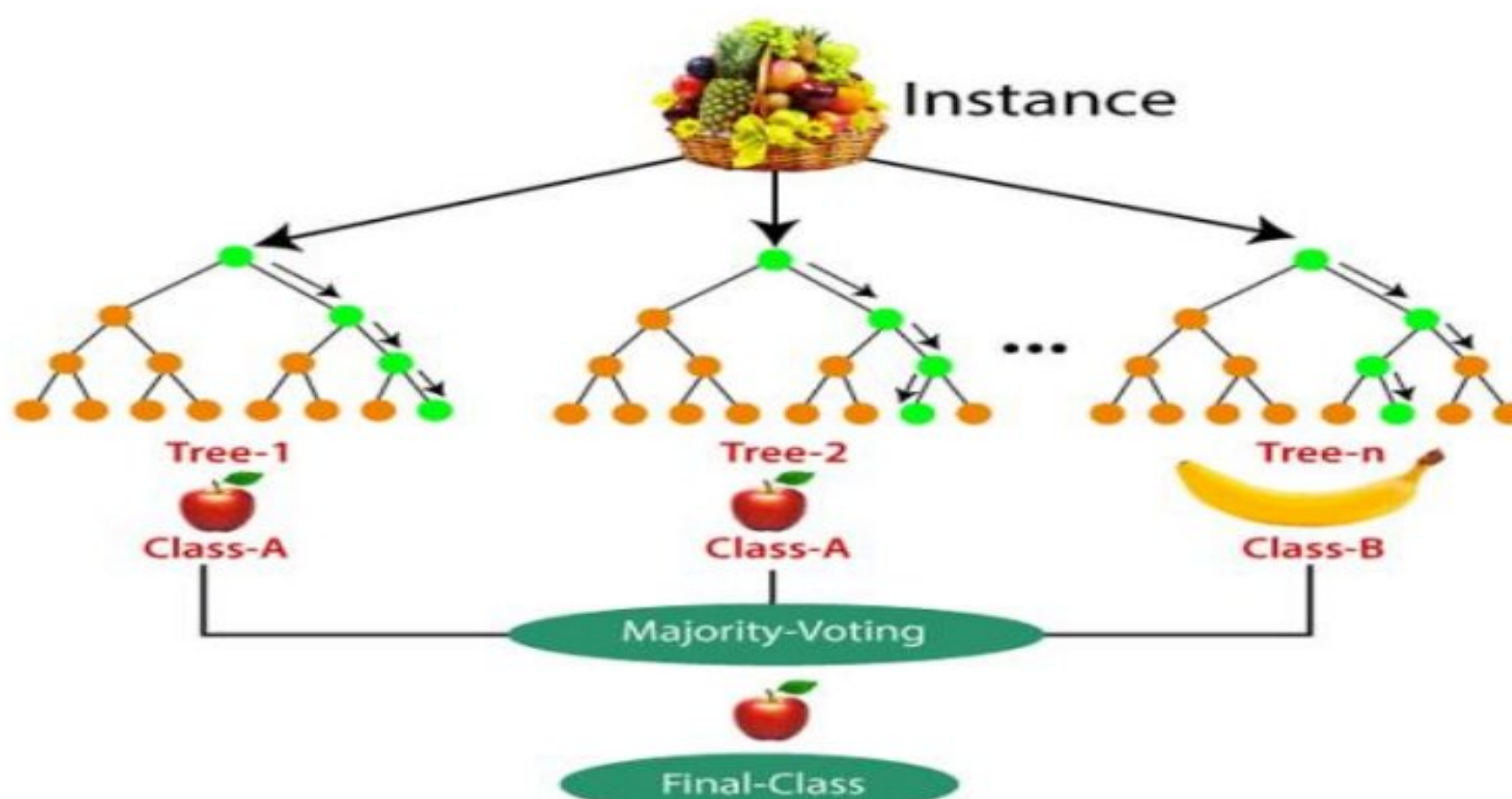
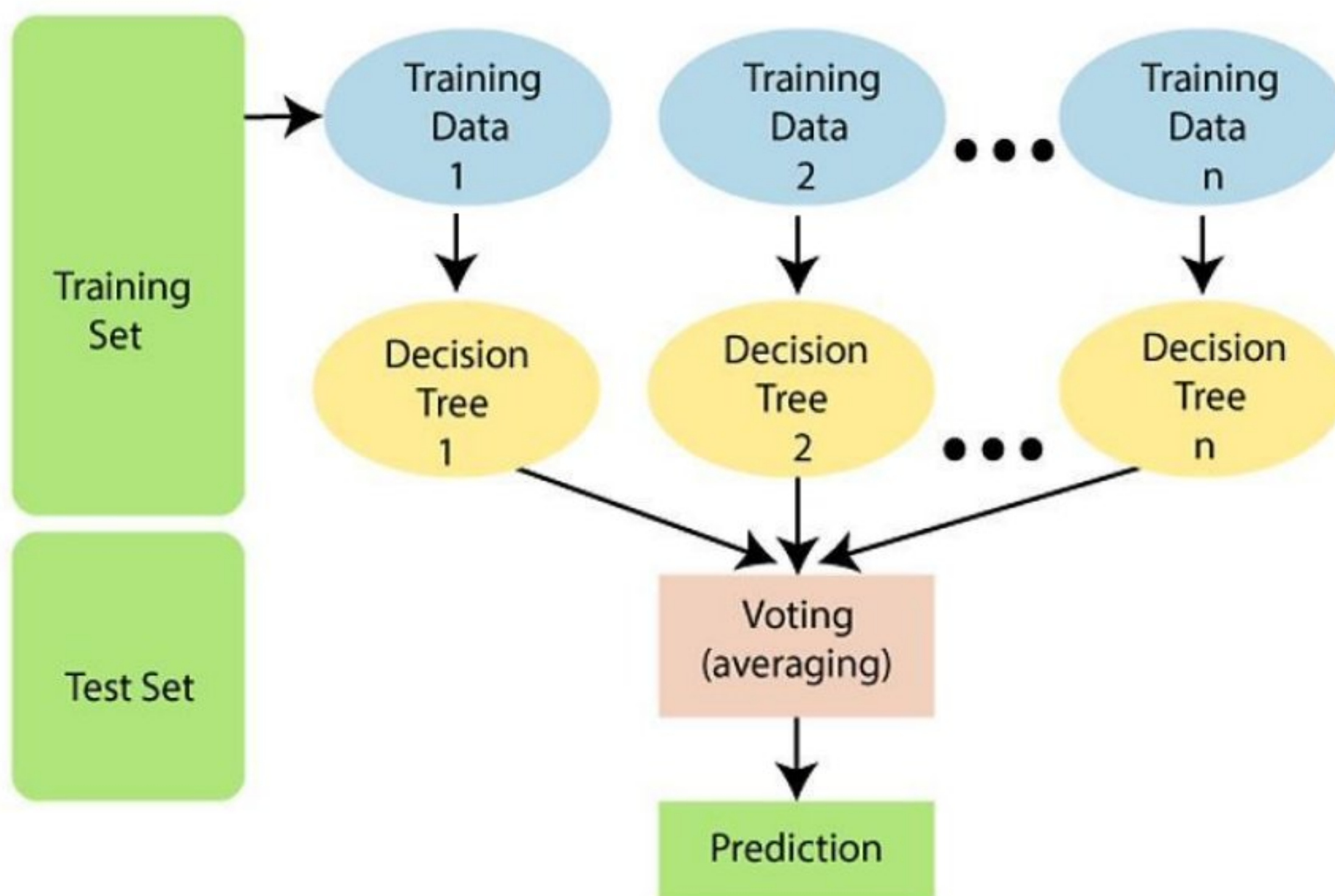


Figure 5.2: Random Forest Classifier Architecture [41]

### Working Procedure:

Step-1: Random 'P' data points are being picked from the given training dataset.

Step-2: Using those chosen data, the decision trees are built.

Step-3: Calculate how many ('D' number of) decision trees you want to construct.

Step-4: Loop on Step 1 & 2.

Step-5: Get majority vote by averaging predictions of from all decision the tree for new data points & that should be the final prediction.

### 5.3 Support Vector Machine [42, 43]

**Introduction:** SVM is regarded as a few of the foremost prevalent supervised algorithms in order to use in Classification or Regression problems. The algorithm portrays the hyperplane (decision boundary) along the n-dimensional training dataset to form different classes so that test data can be put in the appropriate sub-group in the future. SVM uses the extreme points/vectors called support vectors to draw the hyperplane; hence the name Support Vector Machine.

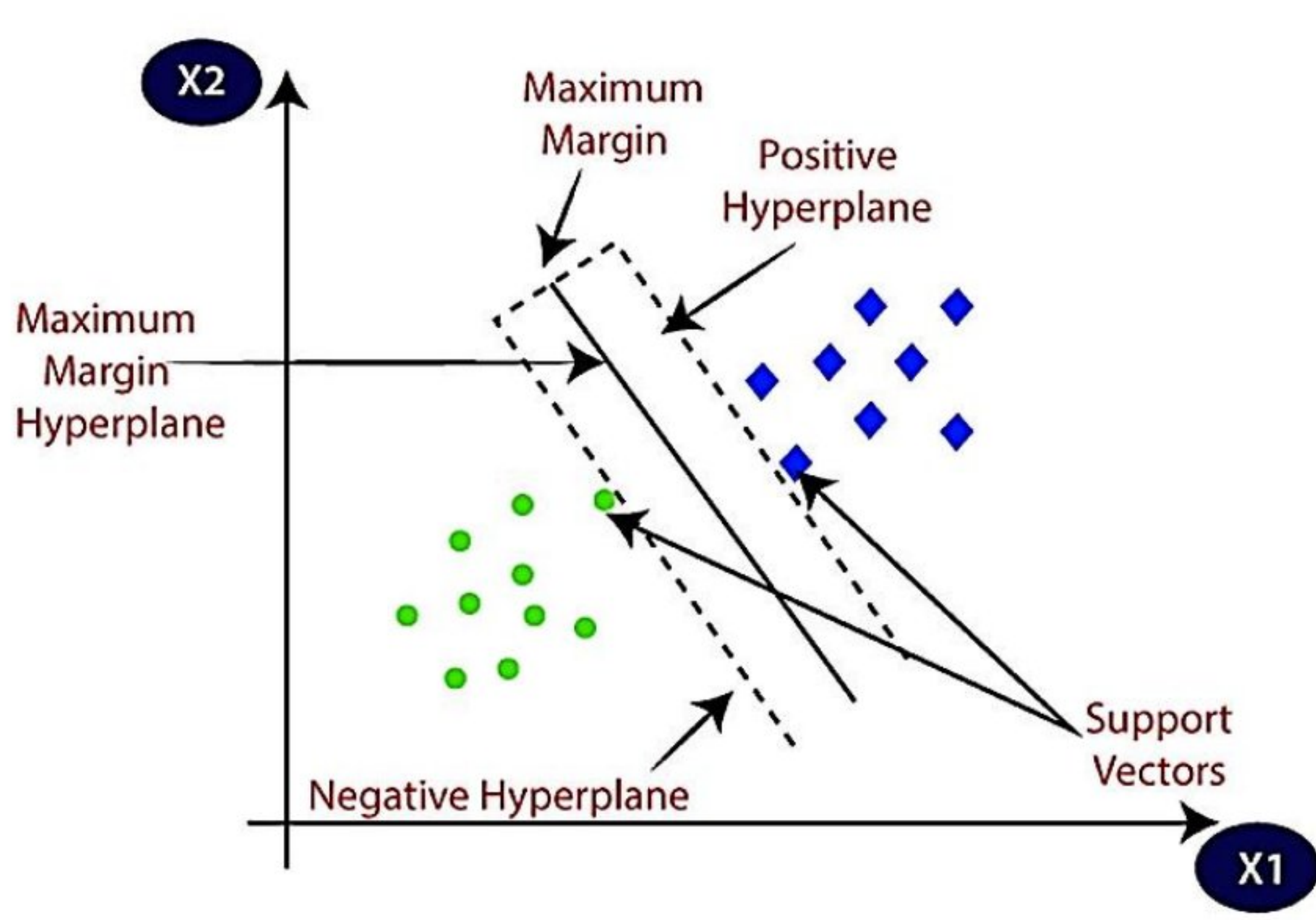
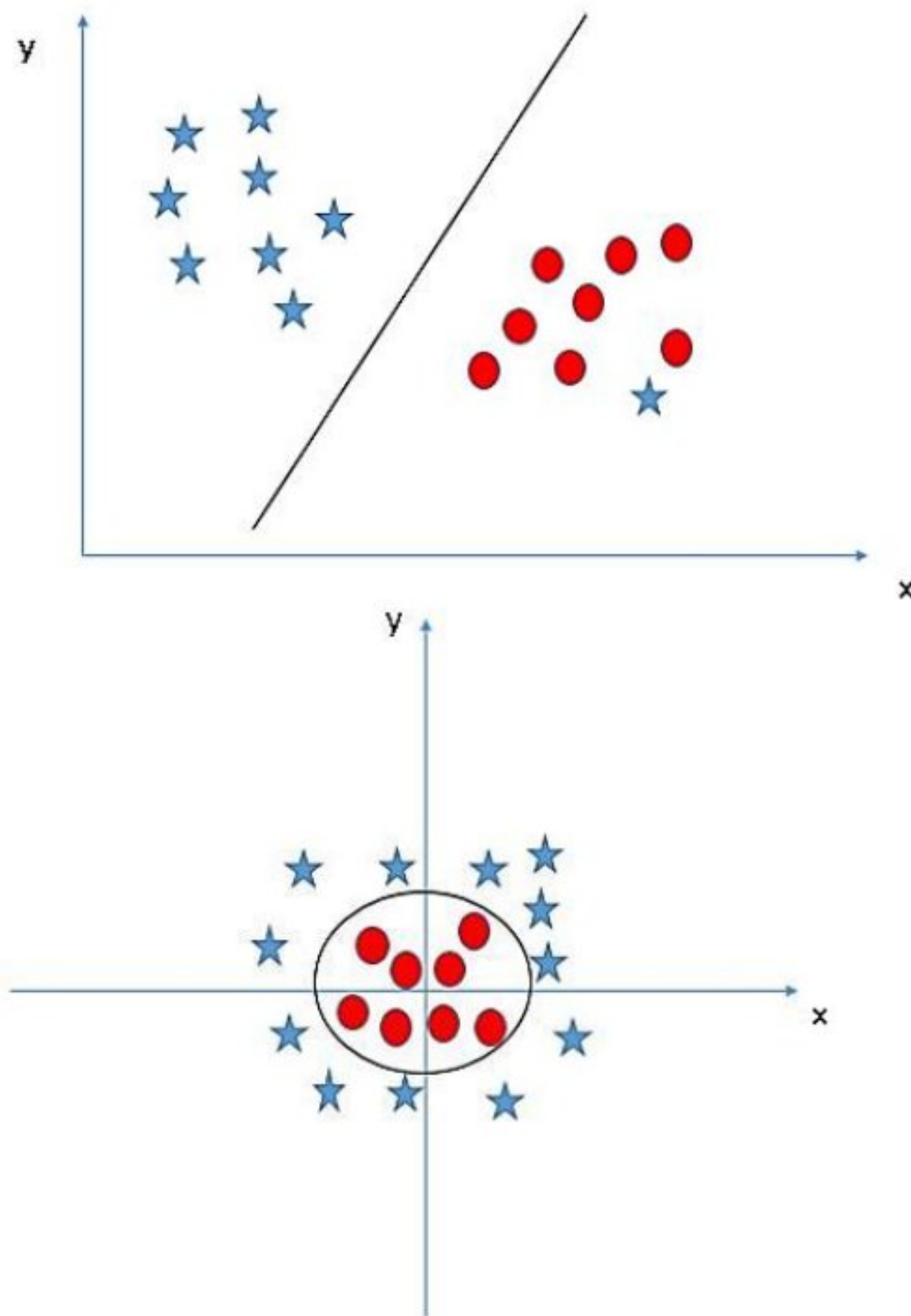


Figure 5.3: Support Vector Machine Architecture [43]

SVM is of 2 types based on hyperplane: Linear SVM & Non-linear SVM



**Figure 5.4:** Linear & Non-linear SVM architecture [43]



**Algorithm:**

Hyperplane on a set of points'  $z'$  can be equated as,

$$\theta^T z - a = 0$$

Hard Margin: It's implemented on a linearly separable training dataset so as to draw two hyperplanes parallelly at the maximum distance separating two classes of data.

$$\text{Hard margin} = \begin{cases} \theta^T z_i - a \geq 1, & \text{if } y_i = 1 \\ \theta^T z_i - a \leq -1, & \text{if } y_i = -1 \end{cases}$$

Optimization problem:

"Minimize  $\|\theta\|$  of  $y_i (\theta^T z_i - a) \geq 1$  where,  $i = 1, \dots, n$ ."

Soft Margin: It's implemented in the case of a non-linearly separable training dataset

Hinge loss function:

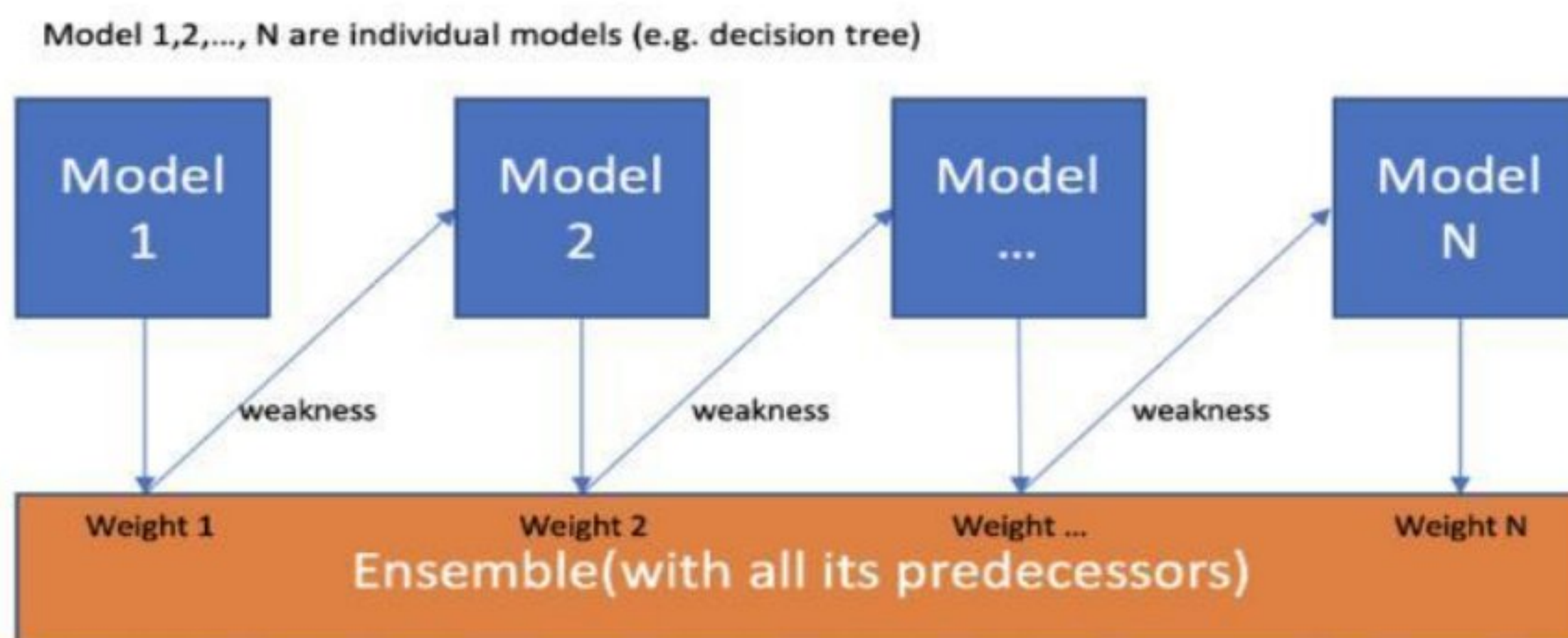
$$\max(0, 1 - y_i (\theta^T z_i - a))$$

The optimization problem is to minimize:

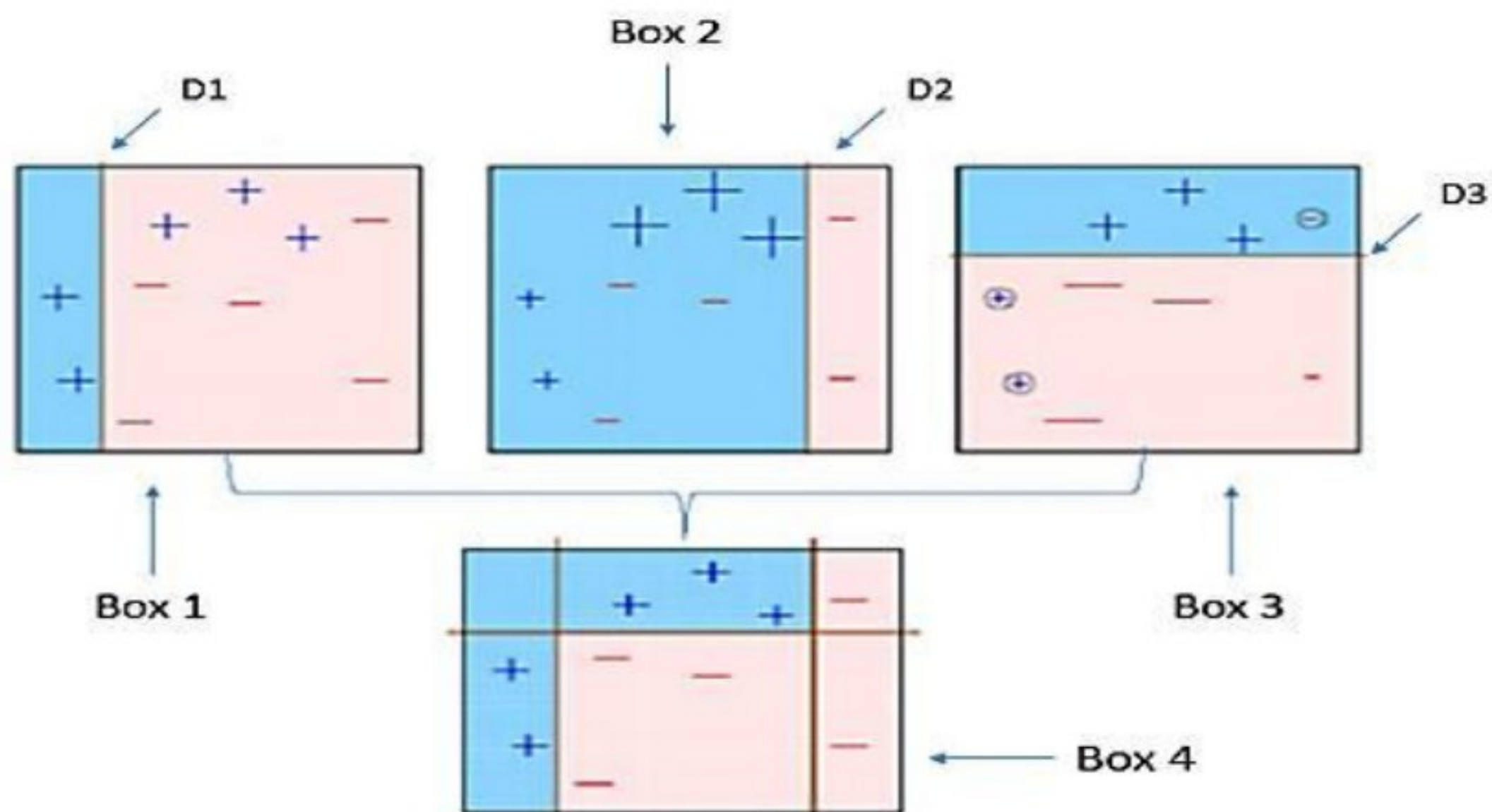
$$\left[ \frac{1}{n} \sum_{i=1}^n \max(0, 1 - y_i (\theta^T z_i - a)) \right] + \lambda \|\theta\|^2$$

## 5.4 AdaBoost [44]

**Introduction:** From several weak classifiers, an ensemble learning technique called Boosting is used in order to develop a robust classifier that also deals with bias-variance trade-offs. While bagging algorithms work for only high variance in a machine learning model, boosting controls both the bias & variance; thus, boosting works more efficiently. Boosting builds models on resampled data to reduce a model's variance to increase its generalization capability. Moreover, boosting works for both generalization and prediction accuracy. Few boosting algorithms can be exemplified as AdaBoost, Gradient Boosting, XGBoost, CatBoost, Light GBM. AdaBoost represents Adaptive Boosting which is the first effective & viable boosting algorithm created to work for binary classification. The tweak in this AdaBoost algorithm is, it gradually ensembles the weak classifiers (with decision trees) after each iteration which previously misclassified some data adjusting the weights to focus on misclassified data & eventually, the ensembled weak classifiers work together as strong classifier with higher accuracy. As a result, AdaBoost is sensitive to noisy data and outliers.



**Fig5.5 : Operating Mechanism of Adaboost [44]**



**Figure 5.6:** AdaBoost Architecture [44]

In the picture above: D1, D2, D3 lines of Box - 1, 2, 3 are weak classifiers trying to separate + & -, but each one does few misclassifications. Together they've become a robust classifier, as stated in Box-4.

**Algorithm:** At every iteration –

- i. Choose a weak classifier  $K_m$
- ii. The classifier  $K_m$  minimizes the total weighted error,

$$\sum_{y_i \neq k_m(x_i)} w_i^{(m)}$$

- iii. Use minimized total weighted error to calculate the error rate,

$$\epsilon_m = \left( \sum_{y_i \neq k_m(x_i)} w_i^{(m)} \right) \cdot \left( \sum_{i=1}^N w_i^{(m)} \right)^{-1}$$

iv. Calculate the weight:

$$\alpha_m = \frac{1}{2} \ln \left( \frac{1 - \epsilon_m}{\epsilon_m} \right)$$

v. Improve the boosted classifier from  $C_{m-1}$  to  $C_m$ :

$$C_m = C_{m-1} + \alpha_m k_m$$

## 5.5 XGBoost [45,46,47]

XGBoost stands for Extreme Gradient Boosting, which uses an optimized distributed gradient boosting (GBM) framework.

Features of XGBoost algorithm:

a) Parallel Computing: This algorithm does parallel processing via using all the cores of a computer processor by default.

b) Regularization: This algorithm performs regularization, which is a technique to avoid overfitting data. It was not available in the GBM framework.

c) Enabled Cross-Validation: XGBoost has an internal CV function, but in many other models, we need to do cross-validation manually.

d) Missing Values: This algorithm can handle missing values internally, even the tendency can be captured by the model.

Model Representation: If  $M$  is the number of trees &  $Q$  is all possible trees, then,

$$\hat{o}_i = \sum_{m=1}^M q_m(x_i) \quad ; q_m \in Q$$

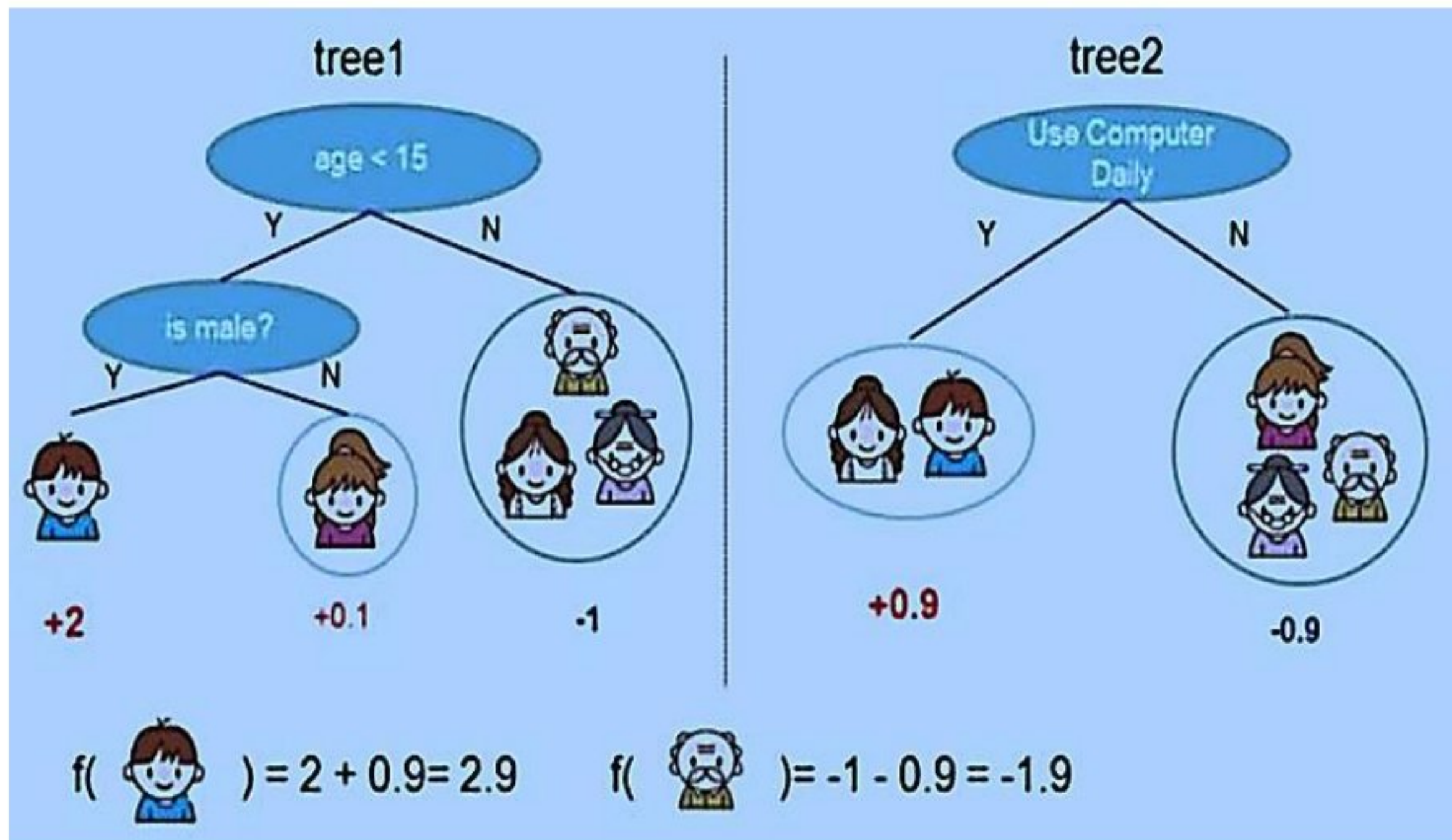


Figure 5.7: XGBoost Architecture [46]

Objective/Loss function:

$$\mathcal{L}(\phi) = \sum_i l(\hat{o}_i, o_i) + \sum_m \Omega(q_m) = \sum_i l(\hat{o}_i, o_i) + \sum_m (\gamma T + \frac{1}{2} \lambda \|\theta\|^2)$$

After regularization & optimization of the above function, the final Objective function becomes:

$$Obj^{(t)} = \sum_{j=1}^T \left[ \left( \sum_{i \in I_j} u_i \right) \theta_j + \frac{1}{2} \left( \sum_{i \in I_j} v_i + \lambda \right) \theta_j^2 \right] + \gamma T$$

where,

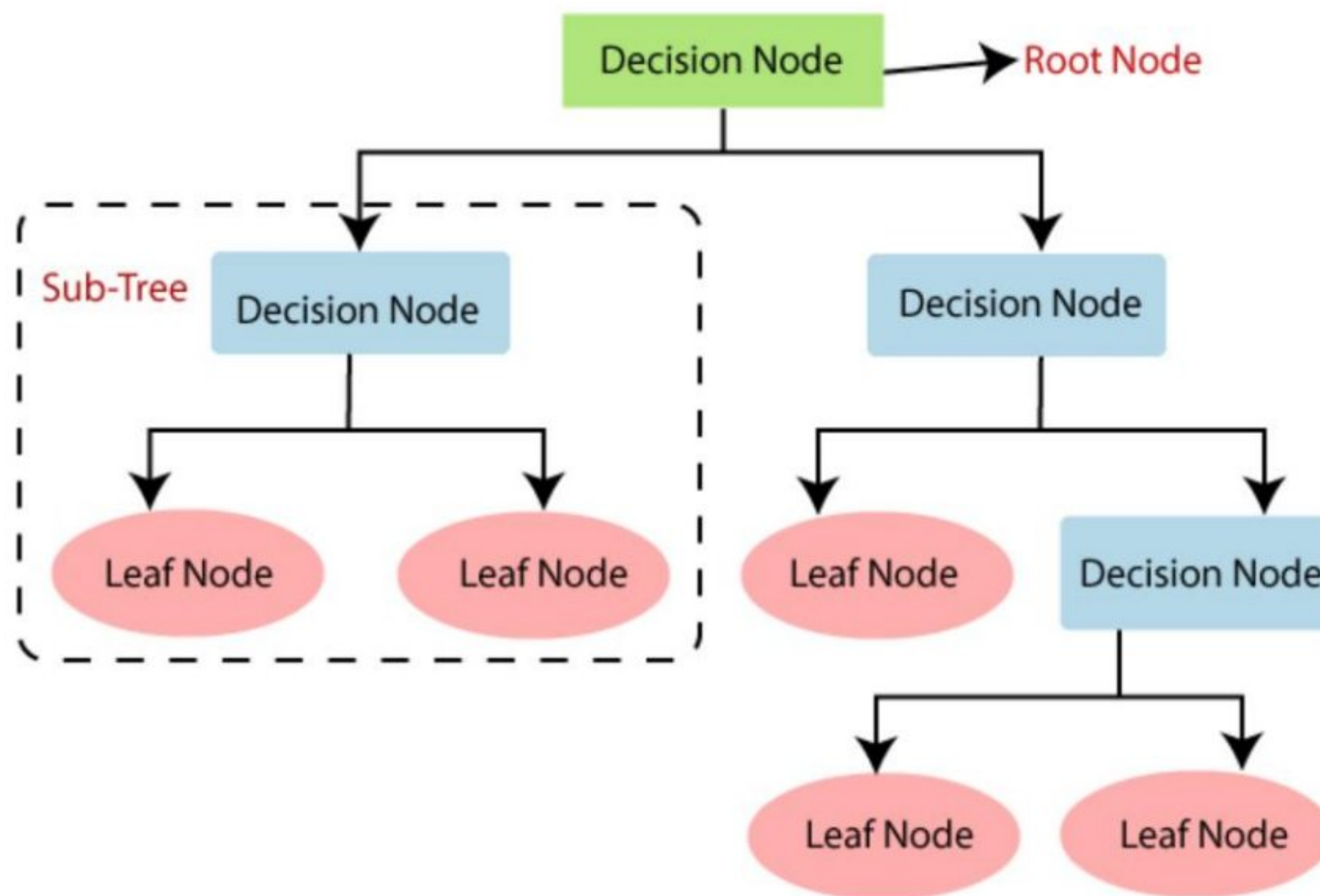
$$Obj^{(t)} = \sum_{j=1}^T \left[ \left( \sum_{i \in I_j} u_i \right) \theta_j + \frac{1}{2} \left( \sum_{i \in I_j} v_i + \lambda \right) \theta_j^2 \right] + \gamma T$$

## ***5.6 Decision Tree Classifier[48]***

### ***Introduction:***

- A supervised learning approach is the Decision Tree method, which may be used to solve regression and classification tasks, though it is most typically employed to solve classification problems.
- Internal nodes are dataset attributes, branches are decision rules, and each leaf node is the conclusion in this tree-structured classifier.
- The Decision Node and the Leaf Node are the two nodes of a Decision Tree. Decision nodes are being used to make decisions and have many branches, while Leaf nodes are the results of such decisions and have no more branches.
- The decisions or tests are made depending on the dataset's attributes.
- It's a pictorial display of all feasible solutions or choice based on a set of criteria.
- This is termed a decision tree because, like a tree, it begins with the root node and grows into a tree-like architecture.
- The CART algorithm (Classification and Regression Tree algorithm) is used to create the tree.
- A decision tree asks a question and then divides it into sub - trees based on the response .

**Architecture:** The basic architecture of Decision Tree Algorithm:



**Fig 5.8: Decision Tree Algorithm structure**

- **Attribute Selection Measures**

The most difficult aspect of creating a Decision tree is deciding which attribute is ideal for the root node and sub-nodes. For addressing these challenges, the Attribute Selection Measure (ASM) mechanism was created. Using this measurement, we can quickly discover the best characteristic for the tree's nodes. Two popular ASM approaches are as follows:

**(I) Information Gain**

**(II) Gini Index**

### (I) Information Gain:

- The assessment of changes in entropy after segmenting a dataset based on an attribute is known as information gain.
- It determines how much data a feature offers about a class.
- We split the node and built the decision tree based on the value of information gained.
- The highest information gain node/attribute is split first in a decision tree method, which always strives to maximize the value of information gain.

The following formula can be used to compute it:

$$\text{Information Gain} = \text{Entropy}(S) - (\text{Weighted Avg}) * \text{Entropy}(\text{each feature})$$

**Entropy:** Entropy is a metric for determining the degree of impurity in a particular property. It defines the randomness of data. Entropy can be estimated using the formula:

$$\text{Entropy}(s) = -P(\text{yes}) \log_2 P(\text{yes}) - P(\text{no}) \log_2 P(\text{no})$$

Where,

- **S= Total number of samples**
- **P(yes)= probability of yes**
- **P(no)= probability of no**

### (II) Gini Index:

- The Gini index is a measure of impurity or purity used in the CART (Classification and Regression Tree) technique to create a decision tree.
- In comparison to a high Gini index, an attribute with a low Gini index should be favoured.
- It only makes binary splits, and the CART method creates binary splits using the Gini index.

The following formula can be used to compute the Gini index:

$$\text{Gini Index} = 1 - \sum_j P_j^2$$



## 5.7 Gaussian Naïve Bayes[49]

**Introduction:** The Bayes Theorem is used to create Naive Bayes Classifiers. The high independence assumptions between the features are one of the assumptions made. These classifiers make the assumption that the value of one feature is unrelated to the value of any other characteristic. Naive Bayes Classifiers are particularly efficient in supervised learning situations. To estimate the parameters needed for classification, naive Bayes classifiers require a small amount of training data. Naive Bayes Classifiers are easy to develop and execute, and they can be used in a variety of real-world settings.

When working with continuous data, one common assumption is that the continuous values associated with each class follow a normal (or Gaussian) distribution. The features' likelihood is considered to be-

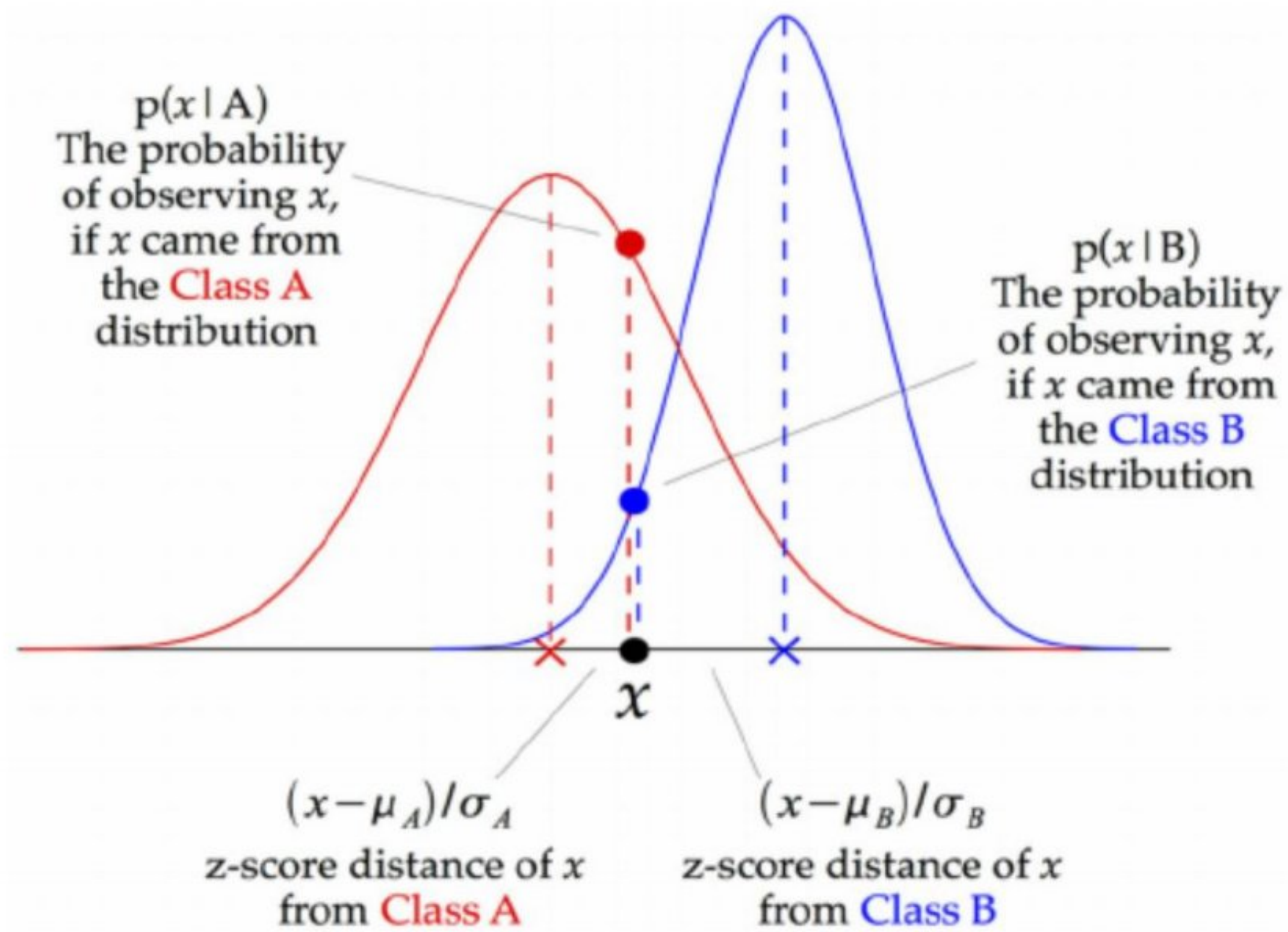
$$P(x_i | y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Sometimes assume variance

- is independent of Y (i.e.,  $\sigma_i$ ),
- or independent of  $X_i$  (i.e.,  $\sigma_k$ )
- or both (i.e.,  $\sigma$ )

Gaussian Naive Bayes accepts continuous valued features and models them all as Gaussian (normal) distributions.

To build a simple model, assume the data is characterized by a Gaussian distribution with no covariance (independent dimensions) between the parameters. This model can be fitted by simply calculating the mean and standard deviation of the points within each label, which is all that is required to construct a distribution of this type.



**Fig 5.9:Class Distribution of GNB in Graphical Representation**

A Gaussian Naive Bayes (GNB) classifier works as shown in the diagram above. The z-score distance between each data point and each class mean is determined, which is the distance from the class mean divided by the class standard deviation.

As a result, we can observe that the Gaussian Naive Bayes technique is slightly different and can be employed well.

## **5.8 CatBoost [50,51]**

CatBoost is a gradient boosting approach based on decision trees. It was developed by Yandex researchers and engineers and is used by Yandex and other companies such as CERN, Cloudflare, and Careem taxi for search, recommendation systems, personal assistants, self-driving cars, weather forecasting, and a variety of other tasks. Because it is open-source, anyone can use it.

The goal of training is to discover the model  $y$  for any input item that best solves the given problem (regression, classification, or multiclassification) based on a set of features  $x_i$ . This model is discovered using a training dataset, which is a collection of objects with known features and label values. The validation dataset is used to validate accuracy. It comprises data in the same format as the training dataset but is only used to assess training quality (it is not used for training).

CatBoost is built on the backbone of gradient enhanced decision trees. During training, a sequence of decision trees is constructed. Each subsequent tree is built with less loss than the ones before it.

The number of trees is decided by the initial parameters. Overfitting should be avoided by using the overfitting detector. When it is enabled, trees stop growing.

### **For a single tree, the steps are as follows:**

The following stages are usually included in the procedure of converting category features to numerical:

- Randomly permuting the set of input objects.
  
- The label value is being converted from a floating point to an integer.
  
- The method is governed by the machine learning task at hand (which is determined by the selected loss function).

The following stages are usually included in the procedure of converting category features to numerical:

- Randomly permuting the set of input objects.
  
- The label value is being converted from a floating point to an integer.
  
- The method is governed by the machine learning task at hand (which is determined by the selected loss function).

**Table 5.1: Performance of Transformation in CatBoost Algorithm**

Problem	How transformation is performed
Regression	Quantization is performed on the label value. The mode and number of buckets ( $k+1$ ) are set in the starting parameters. All values located inside a single bucket are assigned a label value class – an integer in the range $[0;k]$ defined by the formula: $\langle \text{bucket ID} - 1 \rangle$ .
Classification	Possible values for label value are 0 (doesn't belong to the specified target class) and "1" (belongs to the specified target class).
Multiclassification	The label values are integer identifiers of target classes (starting from "0").

Converting category attributes to numerical attributes. The process is determined by the beginning parameters.

## **Types and Formula:**

**Type :** Borders

**Formula:**

Calculating ctr for the  $i$ -th bucket ( $i \in [0; k - 1]$ ):

$$ctr_i = \frac{\text{countInClass} + \text{prior}}{\text{totalCount} + 1}, \text{ where}$$

- `countInClass` is how many times the label value exceeded  $i$  for objects with the current categorical feature value. It only counts objects that already have this value calculated (calculations are made in the order of the objects after shuffling).
- `totalCount` is the total number of objects (up to the current one) that have a feature value matching the current one.
- `prior` is a number (constant) defined by the starting parameters.

**Type :** Buckets

**Formula:**

Calculating ctr for the  $i$ -th bucket ( $i \in [0; k]$ , creates  $k + 1$  buckets):

$$ctr_i = \frac{\text{countInClass} + \text{prior}}{\text{totalCount} + 1}, \text{ where}$$

- `countInClass` is how many times the label value was equal to  $i$  for objects with the current categorical feature value. It only counts objects that already have this value calculated (calculations are made in the order of the objects after shuffling).
- `totalCount` is the total number of objects (up to the current one) that have a feature value matching the current one.
- `prior` is a number (constant) defined by the starting parameters.

**Type :** BinarizedTargetMeanValue

**Formula:**

How ctr is calculated:

$$ctr = \frac{countInClass + prior}{totalCount + 1}, \text{ where}$$

- countInClass is the ratio of the sum of the label value integers for this categorical feature to the maximum label value integer (k).
- totalCount is the total number of objects that have a feature value matching the current one.
- prior is a number (constant) defined by the starting parameters.

**Type :** Counter

**Formula:**

How ctr is calculated for the training dataset:

$$ctr = \frac{curCount + prior}{maxCount + 1}, \text{ where}$$

- curCount is the total number of objects in the training dataset with the current categorical feature value.
- maxCount the number of objects in the training dataset with the most frequent feature value.
- prior is a number (constant) defined by the starting parameters.

How ctr is calculated for the validation dataset:

$$ctr = \frac{curCount + prior}{maxCount + 1}, \text{ where}$$

- The fundamentals of curCount computation vary depending on the calculating method used:
  1. Full — The sum of the total number of objects with the current categorical feature value in the training dataset and the number of objects with the current categorical feature value in the validation dataset.
  2. SkipTest — The total number of items with the current categorical feature value in the training dataset.
  
- maxCount refers to the number of objects having the most frequent feature value in one of the following sets, depending on the computation method used:
  1. Full — Includes both the training and validation datasets.
  2. SkipTest — The training dataset.
  
- **The following approaches for detecting overfitting are supported:**
  - I. IncToDec
  - II. Iter

**I. IncToDec**

CatBoost examines the resulting loss change on the validation dataset before creating each new tree. If the Threshold value set in the starting parameters is greater than CurrentPValue, the overfit detector is triggered;

**CurrentPValue < Threshold**
  
- CurrentPValue is calculated from a set of values for the maximizing metric score[i]:

1. *ExpectedInc* is calculated:

$$ExpectedInc = \max_{i_1 \leq i_2 \leq i} 0.99^{i-i_1} \cdot (score[i_2] - score[i_1])$$

2.  $x$  is calculated:

$$x = \frac{ExpectedInc[i]}{\max_{j \leq i} score[j] - score[i]}$$

3. *CurrentPValue* is calculated:

$$CurrentPValue = \exp\left(-\frac{0.5}{x}\right)$$

## II. Iter

CatBoost counts how many iterations have passed since the last iteration with the best loss function value before starting a new tree.

If the number of iterations is greater than the value indicated in the training parameters, the model is deemed overfitted.

## 5.9 Multi-Layer Perceptron [52]

A multilayer perceptron (MLP) is an artificial neural network that is completely connected and feedforward (ANN). MLP refers to networks constructed up of multiple layers of perceptrons (with threshold activation) in certain settings, and any feedforward ANN in others; see Terminology. Multilayer perceptrons are commonly referred to as "vanilla" neural networks when only one hidden layer is present. [1]

The input layer, the hidden layer, and the output layer are all present in an MLP. Each node, with the exception of the input nodes, is a neuron with a nonlinear activation function. During training, MLP uses backpropagation as a supervised learning technique. [2] [3] MLP is distinguished from a linear perceptron by its numerous layers and non-linear activation. It can discriminate between non-linearly separable data. [4]



- **The Activation function**

If each neuron in a multilayer perceptron has a linear activation function that converts weighted inputs to outputs, then linear algebra shows that any number of layers may be reduced to a two-layer input-output model. A nonlinear activation function is used by some MLP neurons to mimic the frequency of biological neurons' action potentials, or firing.

In history, sigmoids have been the most common activation functions, such as

$$y(v_i) = \tanh(v_i) \quad \text{and} \quad y(v_i) = (1 + e^{-v_i})^{-1}.$$

The rectifier linear unit (ReLU) is being employed increasingly frequently in recent deep learning breakthroughs as one of the possible techniques to handle numerical issues linked to sigmoids.

A hyperbolic tangent has values ranging from -1 to 1, whereas a logistic function has values ranging from 0 to 1. The weighted total of the input connections is represented as  $v$  while the output of the neuron is displayed as  $y$ . As alternative activation functions, the functionalities of the rectifier and softplus have been demonstrated.

- **Layers**

The MLP is made up of three or more layers, one or more hidden layers of nonlinearly activating nodes, and an input and output layer. Because MLPs are completely connected, one layer interacts the next layer according to a weight  $w_{ij}$ .

- **Learning**

After each piece of input is processed, the perceptron adjusts connection weights based on the degree of error in the output relative to the projected result. In this supervised learning example, backpropagation is employed, which is the linear perceptron of an extension of the least-squares technique.

In the  $n$ th data point (training example), the degree of error in an output node  $j$  can be represented by

$$e_j(n) = d_j(n) - y_j(n)$$

Where  $d$  is the preferred measure and  $y$  is the estimated value of the perceptron. Based on the adjustments, the node weights can be modified to reduce overall output inaccuracy, given by,

$$\mathcal{E}(n) = \frac{1}{2} \sum_j e_j^2(n).$$

Gradient descent is used to compute the difference weight,

$$\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial v_j(n)} y_i(n)$$

Here,  $y_i$  is the previous neuron's output, and  $\eta$  is the learning rate, which is chosen to ensure that the weights converge fast and without oscillations.

The induced local field  $v_j$ , which fluctuates, determines the derivative to be computed. It is simple to demonstrate that this derivative can be simplified for an output node.

$$-\frac{\partial \mathcal{E}(n)}{\partial v_j(n)} = e_j(n) \phi'(v_j(n))$$

where  $\phi'$  is the non-fluctuating activation function's derivative.

Even when the required derivative can be demonstrated, analyzing a change in weights to a hidden node is more complex,

$$-\frac{\partial \mathcal{E}(n)}{\partial v_j(n)} = \phi'(v_j(n)) \sum_k -\frac{\partial \mathcal{E}(n)}{\partial v_k(n)} w_{kj}(n).$$

The change in weights of the  $k$ th nodes, which represent the output layer, determines this. The output layer weights fluctuate according to the derivative of the activation function to adjust the hidden layer weights; consequently, this algorithm is a backpropagation of the activation function. [5]

## 5.10 Gradient Boosting Classifier[53]

When the target column is binary, a gradient-boosting classifier is applied. All steps from the Gradient boosting regressor are employed here, with the exception of the loss function. When the target column was continuous, we utilized Mean Squared Error as our loss function; however, this time we'll use log-likelihood.

The loss function for the classification problem is as follows:

$$L = - \sum_{i=1}^n y_i \log(p) + (1 - y_i) \log(1 - p)$$

The initial step in the gradient boosting technique was to set a constant value for the model; previously, we previously used the target column's average, but now we'll use log (odds).

When we differentiate this loss function, we get a log(odds) function, and we need to find a log(odds) number where the loss function is the smallest.

To begin, convert this loss function into a log function (odds)

$$\begin{aligned} L &= - \left[ \sum_{i=1}^n y_i \log(p) + (1 - y_i) \log(1 - p) \right] \\ &= -y * \log(p) - (1 - y) * \log(1 - p) \\ &= -y * \log(p) - \log(1 - p) + y * \log(1 - p) \\ &= y * \left[ \log(p) - \log(1 - p) \right] - \log(1 - p) \\ &= -y * \left[ \frac{\log(p)}{\log(1 - p)} \right] - \log(1 - p) \\ &= -y * \log\left(\frac{p}{1 - p}\right) - \log(1 - p) \end{aligned}$$

$$= -y * \log(\text{odds}) - \log(1 - p)$$

$$\text{Now, } \log(1 - p) = \log\left(1 - \frac{e^{\log(\text{odds})}}{1 + e^{\log(\text{odds})}}\right)$$

$$= \log\left(\frac{1 + e^{\log(\text{odds})}}{1 + e^{\log(\text{odds})}} - \frac{e^{\log(\text{odds})}}{1 + e^{\log(\text{odds})}}\right) = \log\left(\frac{1}{1 + e^{\log(\text{odds})}}\right)$$

$$= \log(1) - \log\left(1 + e^{\log(\text{odds})}\right)$$

$$= 0 - \log\left(1 + e^{\log(\text{odds})}\right)$$

$$\text{Hence, } -y * \log(\text{odds}) - \left(-\log\left(1 + e^{\log(\text{odds})}\right)\right)$$

$$L = -y * \log(\text{odds}) + \log\left(1 + e^{\log(\text{odds})}\right)$$

This is our loss function, and we want to keep it as low as possible. To accomplish so, we set the derivative of it to 0 in relation to  $\log(\text{odds})$ .

$$\frac{dL}{d[\log(\text{odds})]} = -y + \frac{e^{\log(\text{odds})}}{1 + e^{\log(\text{odds})}}$$

We know  $\frac{e^{\log(\text{odds})}}{1 + e^{\log(\text{odds})}} = p$ , hence we can substitute  $p$

$$\frac{dL}{d[\log(\text{odds})]} = -y + p$$

The observed values are denoted by the character  $y$ .

Actually, the function of  $\log(\text{odds})$  and the function of predicted probability " $p$ " are both occasionally simple to use.

The loss function was changed to make computations easier, however it was not necessary.

As a result, Our initial forecast will be the loss function's lowest value .

The fake residuals were created by multiplying the loss function's derivative by -1 in the Gradient boosting regressor. We'll do things the same way we did before. However, the loss function will be dissimilar this time, and we will be dealing with the probability of a result.

$$\frac{dL}{d[\log(\text{odds})]} = -y + p$$

$$\frac{dL}{d[\log(\text{odds})]} = -(-y + p) = (y - p) = (\text{observed} - \text{predicted})$$

We can design a decision tree with all independent variables and target variables as "Residuals" for computing the residuals.

Our first decision tree is now complete, we need to figure out what the leaves' final output value will be because a leaf may get several residuals. the direct formula is simply provided for determining a leaf's output because the math underlying it is outside the scope of this post:

$$\gamma = \frac{\sum_{i=1}^n \text{Residual}_i}{\sum_{i=1}^n [\text{Previous probability}_i \times (1 - \text{Previous probability}_i)]}$$

Finally, fresh predictions may be obtained by integrating the underlying model with the newly constructed residuals tree.

## ***5.11 XG Boost with Random Forest [54]***

A fast gradient boosting implementation for random forest ensemble training is included in the XGBoost package.

The gradient boosting problem is more difficult to solve than the random forest problem. The XGBoost library, which repurposes and utilizes the library's computational efficiencies for random forest model training, can be used to train models.

Extreme Gradient Boosting, or XGBoost for short, is a free and open-source toolkit that implements a more efficient version of the gradient boosting ensemble approach.

As a result, XGBoost is used to refer to the project, library, and algorithm. Gradient boosting is a common approach for classification and regression predictive modeling projects because it consistently gives the best results. Gradient boosting has the drawback of being exceedingly slow to train a model, which is compounded by huge datasets.

XGBoost solves the speed problems with gradient boosting by presenting a set of techniques that considerably speed up model training while simultaneously enhancing overall model performance in a variety of contexts.

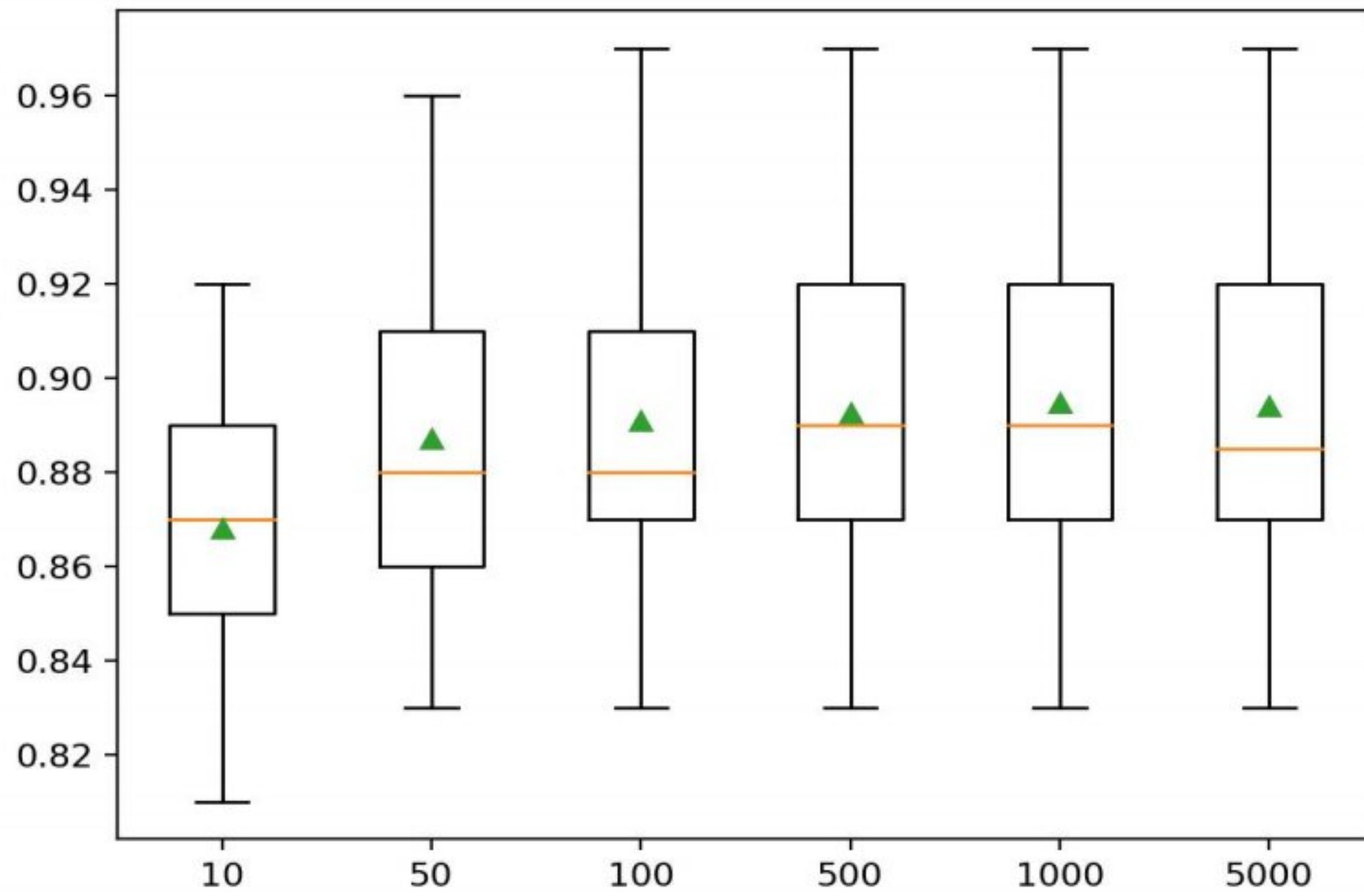
The basic XGBoost strategy can be expanded to include alternative tree ensemble techniques, such as random forest, in addition to gradient boosting.

A combination of decision tree algorithms make form the random forest algorithm.

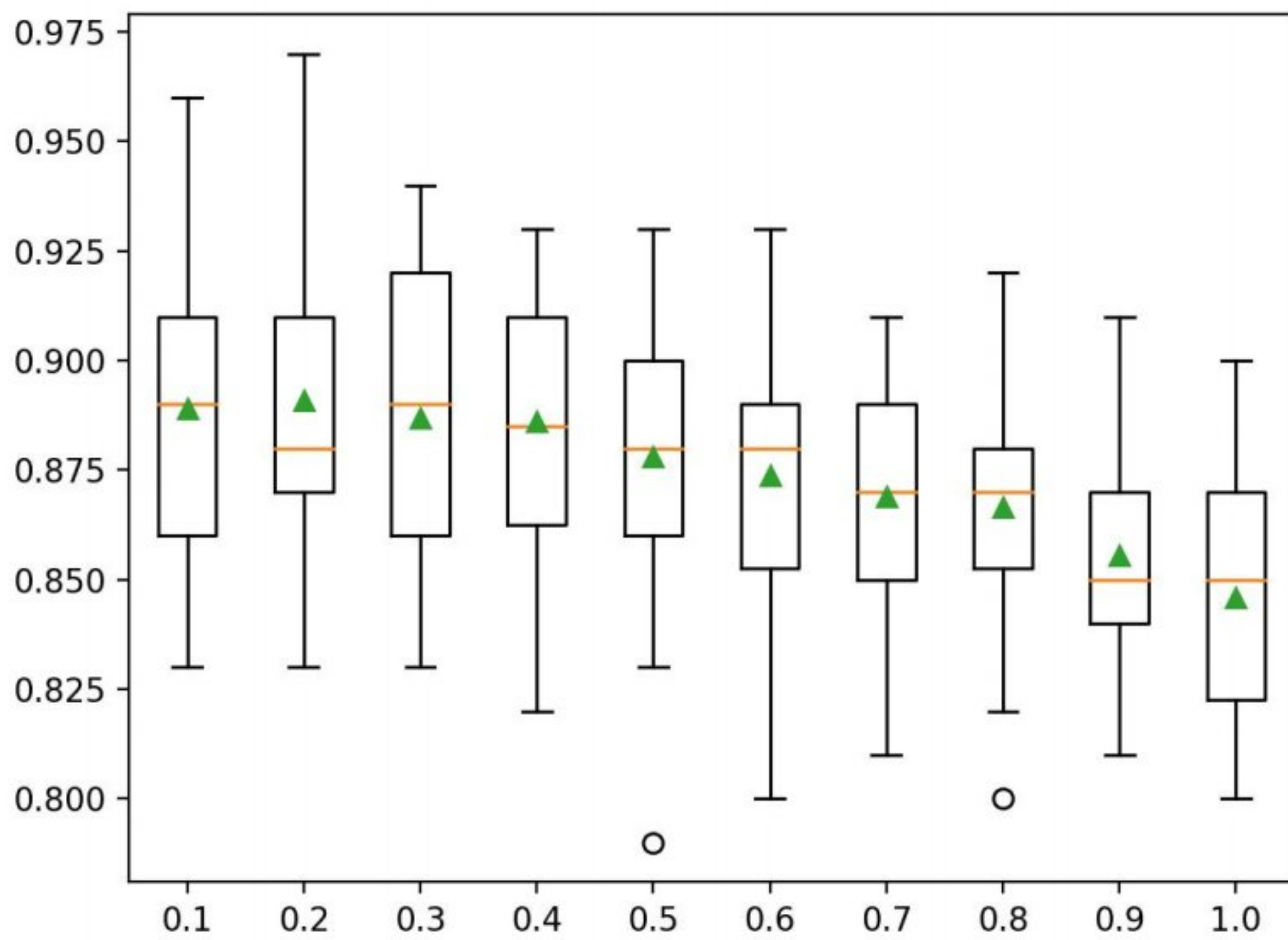
Each decision tree is fitted using a bootstrap sample of the training dataset. This is a sample of the training dataset in which a single example (row) may be selected numerous times, a technique known as sampling with replacement.

A random subset of the input variables (columns) is taken into account at each split point in the tree. As a result, each tree in the ensemble is not just talented, but also distinct in some way. A small selection of features are typically assessed at each split point. When dealing with classification difficulties, a popular heuristic is to choose a number of features equal to the square root of the entire number of features, such as 4 if a dataset comprises 20 input variables.

To represent the distribution of accuracy scores, a box and whisker plot is constructed for each configured number of trees.



**Fig 5.10: A Box and whisker plot of XGBRF Ensemble Size vs. Accuracy Assessment**



**Fig 5.11: A Box and whisker plot of XGBRF Feature Set Size vs. Accuracy Assessment**

# Chapter 6

## RESULT & ANALYSIS

### **6.1 Implementation of Machine Learning Algorithms**

Eleven machine learning algorithms were implemented to diagnose cervical cancer. Firstly, they were implemented with the default hyperparameter (DHP) setting. The performance was too low. That is why Grid Search Cross-Validation (GSCV) hyperparameter tuning technique was introduced. Grid Search Cross-Validation (GSCV) is an efficient hyperparameter tuning technique that is suitable for a large amount of training data. Every machine learning model was trained with the GSCV hyperparameter optimization technique, and the performance of the model was increased.

#### **6.1.1 K-Nearest Neighbor**

**Table 6.1 Confusion Matrix (Without Hyperparameter Tuning) (KNN)**

Confusion Matrix (KNN)	True	False
Positive	9	2
Negative	3	1

**Table 6.2 Performance Matrix (Without Hyperparameter Tuning) (KNN)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
KNN	0.8	0.8182	0.8571	0.9	0.7841

**Table 6.3 Confusion Matrix (With Hyperparameter Tuning) (KNN)**

Confusion Matrix (KNN)	True	False
Positive	10	1
Negative	4	0

**Table 6.4 Performance Matrix (With Hyperparameter Tuning) (KNN)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
KNN	0.9333	0.9091	0.9524	1.0	0.9545



## 6.1.2 Random Forest Classifier

**Table 6.5 Confusion Matrix (Without Hyperparameter Tuning) (RFC)**

<b>Confusion Matrix (RFC)</b>	True	False
Positive	10	1
Negative	3	1

**Table 6.6 Performance Matrix (Without Hyperparameter Tuning) (RFC)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
RFC	0.8667	0.9091	0.9091	0.9091	0.8295

**Table 6.7 Confusion Matrix (With Hyperparameter Tuning) (RFC)**

<b>Confusion Matrix (RFC)</b>	True	False
Positive	10	1
Negative	4	0

**Table 6.8 Performance Matrix (With Hyperparameter Tuning) (RFC)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
RFC	0.9333	0.9091	0.9524	1.0	0.9545

## 6.1.3 Support Vector Machine

**Table 6.9 Confusion Matrix (Without Hyperparameter Tuning) (SVM)**

<b>Confusion Matrix (SVM)</b>	True	False
Positive	10	1
Negative	3	1

**Table 6.10 Performance Matrix (Without Hyperparameter Tuning) (SVM)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
SVM	0.8667	0.9091	0.9091	0.9091	0.8295

**Table 6.11 Confusion Matrix (With Hyperparameter Tuning) (SVM)**

<b>Confusion Matrix (SVM)</b>	True	False
Positive	10	1
Negative	4	0

**Table 6.12 Performance Matrix (With Hyperparameter Tuning) (SVM)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
SVM	0.9333	0.9091	0.9524	1.0	0.9545

### **6.1.4 AdaBoost**

**Table 6.13 Confusion Matrix (Without Hyperparameter Tuning) (AdaB)**

Confusion Matrix (AdaB)	True	False
Positive	9	2
Negative	3	1

**Table 6.14 Performance Matrix (Without Hyperparameter Tuning) (AdaB)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
AdaB	0.8	0.8182	0.8571	0.9	0.7841

**Table 6.15 Confusion Matrix (With Hyperparameter Tuning) (AdaB)**

Confusion Matrix (AdaB)	True	False
Positive	9	2
Negative	3	1

**Table 6.16 Performance Matrix (With Hyperparameter Tuning) (AdaB)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
AdaB	0.8	0.8182	0.8571	0.9	0.7841

### **6.1.5 XGBoost**

**Table 6.17 Confusion Matrix (Without Hyperparameter Tuning) (XGB)**

Confusion Matrix (XGB)	True	False
Positive	9	2
Negative	3	1

**Table 6.18 Performance Matrix (Without Hyperparameter Tuning) (XGB)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
XGB	0.8	0.8182	0.8571	0.9	0.7841

**Table 6.19 Confusion Matrix (With Hyperparameter Tuning) (XGB)**

Confusion Matrix (XGB)	True	False
Positive	9	2
Negative	3	1

**Table 6.20 Performance Matrix (With Hyperparameter Tuning) (XGB)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
XGB	0.8	0.8182	0.8571	0.9	0.7841

## 6.1.6 Decision Tree Classifier

**Table 6.21 Confusion Matrix (Without Hyperparameter Tuning) (DTC)**

Confusion Matrix (DTC)	True	False
Positive	9	2
Negative	3	1

**Table 6.22 Performance Matrix (Without Hyperparameter Tuning) (DTC)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
DTC	0.8	0.8182	0.8571	0.9	0.7841

**Table 6.23 Confusion Matrix (With Hyperparameter Tuning) (DTC)**

Confusion Matrix (DTC)	True	False
Positive	10	1
Negative	4	0

**Table 6.24 Performance Matrix (With Hyperparameter Tuning) (DTC)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
DTC	0.9333	0.9091	0.9524	1.0	0.9545

## 6.1.7 Gaussian Naïve Bayes

**Table 6.25 Confusion Matrix (Without Hyperparameter Tuning) (GNB)**

Confusion Matrix (GNB)	True	False
Positive	9	2
Negative	4	0

**Table 6.26 Performance Matrix (Without Hyperparameter Tuning) (GNB)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
GNB	0.8667	0.8182	0.9	1.0	0.9091

**Table 6.27 Confusion Matrix (With Hyperparameter Tuning) (GNB)**

Confusion Matrix (GNB)	True	False
Positive	9	2
Negative	4	0

**Table 6.28 Performance Matrix (With Hyperparameter Tuning) (GNB)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
GNB	0.8667	0.8182	0.9	1.0	0.9091

## 6.1.8 CatBoost

**Table 6.29 Confusion Matrix (Without Hyperparameter Tuning) (CatB)**

Confusion Matrix (CatB)	True	False
Positive	9	2
Negative	4	0

**Table 6.30 Performance Matrix (Without Hyperparameter Tuning) (CatB)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
CatB	0.8667	0.8182	0.9	1.0	0.9091

**Table 6.31 Confusion Matrix (With Hyperparameter Tuning) (CatB)**

Confusion Matrix (CatB)	True	False
Positive	9	2
Negative	3	1

**Table 6.32 Performance Matrix (With Hyperparameter Tuning) (CatB)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
CatB	0.8	0.8182	0.8571	0.9	0.7841

## 6.1.9 Multi-Layer Perceptron

**Table 6.33 Confusion Matrix (Without Hyperparameter Tuning) (MLP)**

Confusion Matrix (MLP)	True	False
Positive	10	1
Negative	4	0

**Table 6.34 Performance Matrix (Without Hyperparameter Tuning) (MLP)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
MLP	0.9333	0.9091	0.9524	1.0	0.8636

**Table 6.35 Confusion Matrix (With Hyperparameter Tuning) (MLP)**

Confusion Matrix (MLP)	True	False
Positive	10	1
Negative	4	0

**Table 6.36 Performance Matrix (With Hyperparameter Tuning) (MLP)**

Algorithms	Accuracy	Precision	F1	Recall	ROC_AUC
MLP	0.9333	0.9091	0.9524	1.0	0.8636

### 6.1.10 Gradient Boosting Classifier

**Table 6.37 Confusion Matrix (Without Hyperparameter Tuning) (GradB)**

<b>Confusion Matrix (GradB)</b>	True	False
Positive	8	3
Negative	4	0

**Table 6.38 Performance Matrix (Without Hyperparameter Tuning) (GradB)**

<b>Algorithms</b>	<b>Accuracy</b>	<b>Precision</b>	<b>F1</b>	<b>Recall</b>	<b>ROC_AUC</b>
GradB	0.8	0.7273	0.8421	1.0	0.9545

**Table 6.39 Confusion Matrix (With Hyperparameter Tuning) (GradB)**

<b>Confusion Matrix (GradB)</b>	True	False
Positive	8	3
Negative	4	0

**Table 6.40 Performance Matrix (With Hyperparameter Tuning) (GradB)**

<b>Algorithms</b>	<b>Accuracy</b>	<b>Precision</b>	<b>F1</b>	<b>Recall</b>	<b>ROC_AUC</b>
GradB	0.8	0.7273	0.8421	1.0	0.9545

### 6.1.11 XG Boost with Random Forest

**Table 6.41 Confusion Matrix (Without Hyperparameter Tuning) (XGBRF)**

<b>Confusion Matrix (XGBRF)</b>	True	False
Positive	10	1
Negative	3	1

**Table 6.42 Performance Matrix (Without Hyperparameter Tuning) (XGBRF)**

<b>Algorithms</b>	<b>Accuracy</b>	<b>Precision</b>	<b>F1</b>	<b>Recall</b>	<b>ROC_AUC</b>
XGBRF	0.8667	0.9091	0.9091	0.9091	0.8295

***Table 6.43 Confusion Matrix (With Hyperparameter Tuning) (XGBRF)***

<b>Confusion Matrix (XGBRF)</b>	<b>True</b>	<b>False</b>
Positive	10	1
Negative	3	1

***Table 6.44 Performance Matrix (With Hyperparameter Tuning) (XGBRF)***

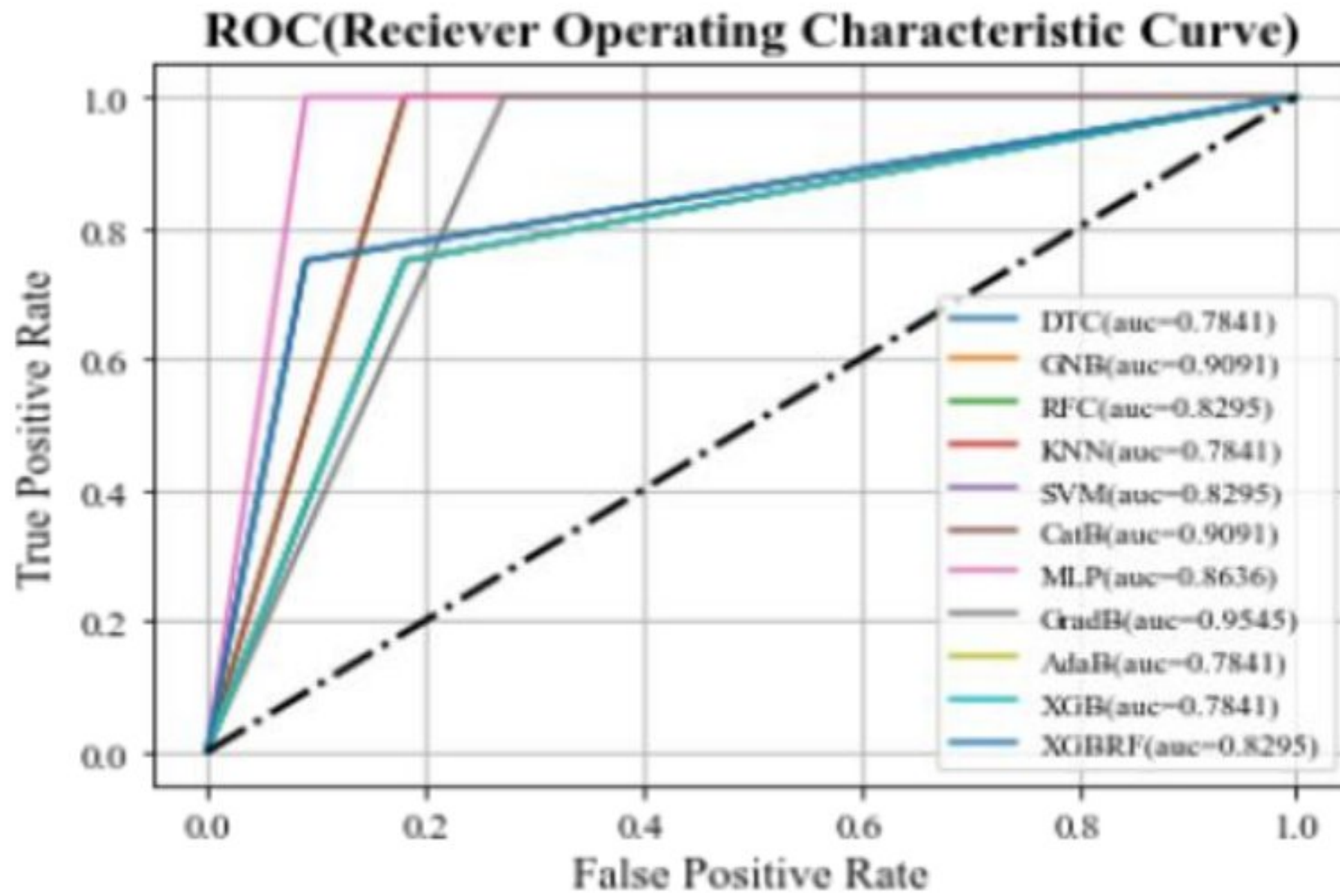
<b>Algorithms</b>	<b>Accuracy</b>	<b>Precision</b>	<b>F1</b>	<b>Recall</b>	<b>ROC_AUC</b>
XGBRF	0.8667	0.9091	0.9091	0.9091	0.8295

## ***6.2 Overall Performance of All the Classifiers:***

**Table 6.45:** Overall Accuracy of Classifiers from Machine Learning ( Without Hyperparameter Tuning)

<b>Classifier</b>	<b>Accuracy</b>
K-Nearest Neighbour	0.8
Random Forest Classifier	0.867
Support Vector Machine	0.867
AdaBoost	0.8
XG Boost	0.8
Decision Tree Classifier	0.8
Gaussian Naive Bayes	0.867
CatBoost	0.867
Multi-Layer Perceptron	0.933
Gradient Boosting Classifier	0.8
XG Boosting with Random Forest Classifier	0.867

- **Receiver Operation Characteristic (ROC) Curve (without Hyperparameter Tuning)**



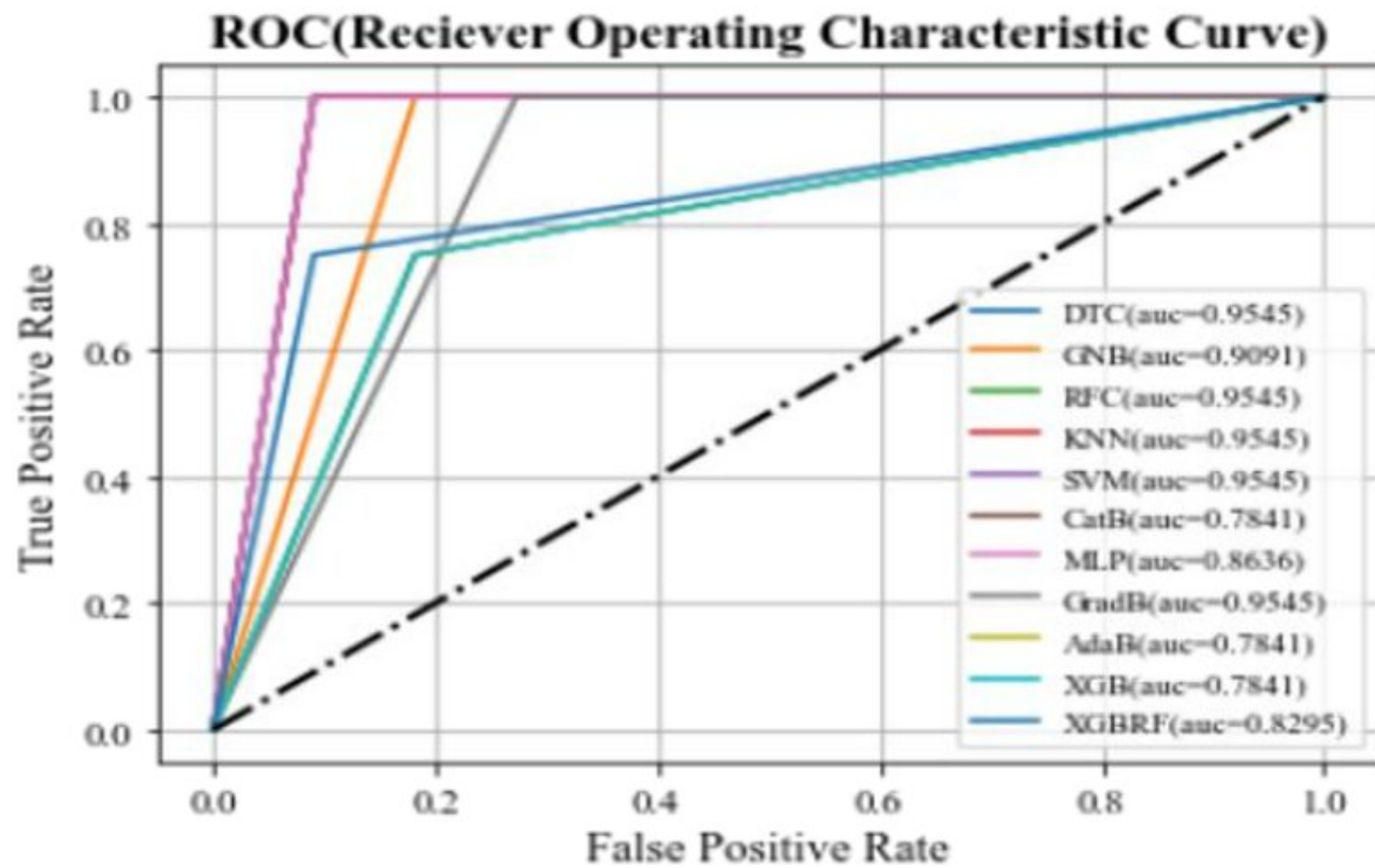
**Fig 6.1: ROC Curve Without Hyperparameter Tuning**

**Table 6.46: Overall Accuracy of Classifiers from Machine Learning ( With Hyperparameter Tuning)**

Classifier	Accuracy
K-Nearest Neighbour	0.9333
Random Forest Classifier	0.9333
Support Vector Machine	0.9333
AdaBoost	0.8
XG Boost	0.8
Decision Tree Classifier	0.9333
Gaussian Naive Bayes	0.8667
CatBoost	0.8
Multi-Layer Perceptron	0.9333
Gradient Boosting Classifier	0.8
XG Boosting with Random Forest Classifier	0.8667



- **Receiver Operation Characteristic (ROC) Curve (with Hyperparameter Tuning)**



**Fig 6.2: ROC Curve With Hyperparameter Tuning**

### 6.3 Analysis:

The ML models are trained with train data and tested with test data using the default hyperparameters of the ML methods, with the performance metrics evaluated and given in Tables. These tables show the confusion matrices, and the Receiver Operation Characteristic Curve (ROC) for this approach. In most performance criteria, the MLP algorithms outperformed all other algorithms.

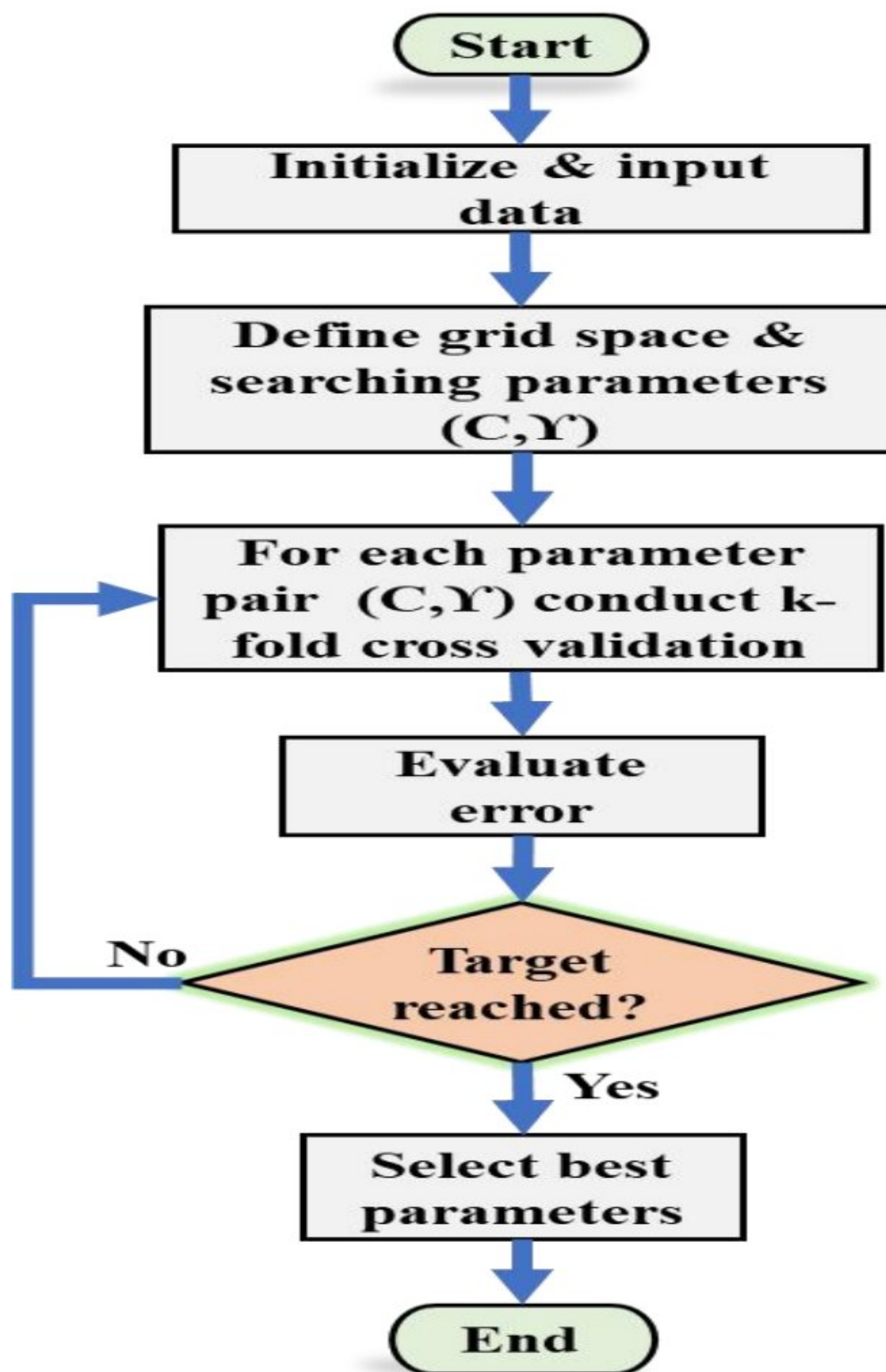


Fig 6.3: Grid Search Algorithm Flow Chart

As can be observed, MLP outperforms all other models in terms of accuracy (0.9333), precision (0.9091), F1 score (0.9524), and recall (1.000). Precision is maximized by RFC, SVM, and XGBRF, whereas recall is maximized by GNB, CatB, and GradB. Furthermore, the maximum ROC\_AUC was 0.9545 when utilizing the GradB algorithm. The performance metrics of ML models were tested in the second approach after hyperparameter tuning was performed using a 10-fold GSCV method.

The confusion matrices, and the ROC curve for this approach. According to the performance measures provided in the Tables, the DTC, RFC, KNN, and SVM beat all ML models in terms of accuracy (0.9333), precision (0.9091), F1 score (0.9524), recall (1.000), and ROC\_AUC (0.9545). GradB, like XGBRF and GNB, maximizes precision and recall.

The results of the experiments are largely good. As can be shown, only one algorithm (MLP) fared best (accuracy) using the default hyperparameters; however, when hyperparameter tuning was done using the GSCV technique, numerous algorithms (DTC, RFC, KNN, SVM, MLP) beat the previous experiment. Furthermore, the second strategy improved the overall performance of ML algorithms. This is a remarkable finding from the study, and it may aid in the development of a more accurate, efficient, and reliable cervical cancer risk prediction algorithm.

As a result, this work proposes an exploratory technique to demonstrate the performance of several machine learning classifiers in risk prediction so that healthcare providers can have a head start and patients with cervical cancer can receive quick diagnosis. As a result, a clever and intelligent support system can be built, as well as an effective healthcare management system, so that people from all walks of life can receive proper cancer treatment.

## **Chapter 7**

### **CONCLUSION & FUTURE SCOPES**

#### ***7.1 Conclusion***

Cervical cancer is one of the most serious health problems that women face around the world. This disease claims the lives of a large number of people every year. As a result, early detection or risk prediction can aid in reducing the number of people who die from this disease. With a significant amount of data, machine learning algorithms are employed to create a reliable prediction model for early detection of cervical cancer risks in women. This study compares the accuracy of eleven supervised machine learning algorithms in predicting cervical cancer risk.

The maximum accuracy found in this study was 93.33% using Decision Tree Classifier (DTC), Random Forest Classifier (RFC), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Multi-Layer Perceptron (MLP) algorithms, and hyperparameter tuning was performed using GSCV to improve the performance of classifiers. The study's key conclusion is that forecast consistency and accuracy have increased.

This discovery could help with the development and implementation of computer-aided diagnosis and serve as a useful tool for healthcare providers. However, before being used in a clinical context, this should be thoroughly tested. Many future advancements in this field can be achieved by collecting more data, which can aid in more accurate estimation and contribute significantly to the creation of an e-healthcare system.

#### ***7.2 Future Scopes***

By including a few non-invasive features such as family disease history, patient demographics, and lifestyle, the suggested model holds the potential for future improvement. On the UCI Repository data set, the proposed model appears to be quite efficient. In the future, the suggested model can be trained with more real-world data from hospitals and diagnostic facilities to reduce the amount of volatility. Training utilizing regionally acquired data would also aid in providing better results to the people of the region.

## References

- [1] “Cervical cancer.” <https://www.who.int/news-room/fact-sheets/detail/cervical-cancer> (accessed Feb. 19, 2022).
- [2] W. H. O. R. Health, W. H. O. C. Diseases, and H. Promotion, *Comprehensive Cervical Cancer Control: A Guide to Essential Practice*, World Health Organization, Geneva, Switzerland, 2006.
- [3] WHO, “Comprehensive Cervical Cancer Control,” *Geneva*, pp. 366–378, 2014
- [4] M. Safaeian, D. Solomon, and P. E. Castle, “Cervical Cancer Prevention-Cervical Screening: Science in Evolution,” *Obstet. Gynecol. Clin. North Am.*, vol. 34, no. 4, pp. 739–760, 2007, doi: 10.1016/j.ogc.2007.09.004.
- [5] N. Colombo, S. Carinelli, A. Colombo, C. Marini, D. Rollo, and C. Sessa, “Cervical cancer: ESMO clinical practice guidelines for diagnosis, treatment and follow-up,” *Ann. Oncol.*, vol. 23, no. SUPPL. 7, 2012, doi: 10.1093/annonc/mds268.
- [6] I. C. Scarinci et al., “Cervical cancer prevention: New tools and old barriers,” *Cancer*, vol. 116, no. 11, pp. 2531–2542, 2010, doi: 10.1002/cncr.25065.
- [7] J. Doorbar, “Molecular biology of human papillomavirus infection and cervical cancer,” *Clin. Sci.*, vol. 110, no. 5, pp. 525–541, 2006, doi: 10.1042/CS20050369.
- [8] D. Saslow et al., “American Cancer Society, American Society for Colposcopy and Cervical Pathology, and American Society for Clinical Pathology screening guidelines for the prevention and early detection of cervical cancer,” *CA. Cancer J. Clin.*, vol. 62, no. 3, pp. 147–172, 2012, doi: 10.3322/caac.21139.
- [9] J. Lu, E. Song, A. Ghoneim, and M. Alrashoud, “Machine learning for assisting cervical cancer diagnosis: An ensemble approach,” *Futur. Gener. Comput. Syst.*, vol. 106, pp. 199–205, 2020, doi: 10.1016/j.future.2019.12.033.
- [10] B. Nithya and V. Ilango, “Evaluation of machine learning based optimized feature selection approaches and classification methods for cervical cancer prediction,” *SN Appl. Sci.*, vol. 1, no. 6, 2019, doi: 10.1007/s42452-019-0645-7.
- [11] R. Weegar and K. Sundström, “Using machine learning for predicting cervical cancer from Swedish electronic health records by mining hierarchical

- representations,” *PLoS One*, vol. 15, no. 8 August 2020, pp. 1–19, 2020, doi: 10.1371/journal.pone.0237911.
- [12] [2] World Cancer Report, World Health Org., Geneva, Switzerland, 2014.
- [13] G. N. Papanicolaou and H. F. Traut, “Diagnosis of Uterine Cancer by the Vaginal Smear”, The Commonwealth Fund, New York, pp. 19–45, 1943.
- [14] P. T. T. Wong, R. K. Wong, T. A. Caputo, T. A. Godwin and B. Rigas, “Infrared Spectroscopy of Exfoliated Human Cervical Cells: Evidence of Extensive Structural Changes During Carcinogenesis”, *Proceedings of the National Academy of Sciences*, Vol. 88, pp. 10988–10992, 1991.
- [15] Exner, M.; Kühn, A.; Stumpp, P.; Höckel, M.; Horn, L.C.; Kahn, T.; Brandmaier, P. Value of diffusion-weighted MRI in diagnosis of uterine cervical cancer: A prospective study evaluating the benefits of DWI compared to conventional MR sequences in a 3T environment. *Acta. Radiol.* 2016, 57, 869–877. [CrossRef]
- [16] McVeigh, P.Z.; Syed, A.M.; Milosevic, M.; Fyles, A.; Haider, M.A. Diffusion-weighted MRI in cervical Cancer. *Eur. Radiol.* 2008, 18, 1058–1064. [CrossRef]
- [17] Wu, W.; Zhou, H. Data-driven diagnosis of cervical cancer with support vector machine-based approaches. *IEEE Access* 2017, 5, 25189–25195. [CrossRef]
- [18] Yang, J.; Nolte, F.S.; Chajewski, O.S.; Lindsey, K.G.; Houser, P.M.; Pellicier, J.; Wang, Q.; Ehsani, L. Cytology and high risk HPV testing in cervical cancer screening program: Outcome of 3-year follow-up in an academic institute. *Diagn. Cytopathol.* 2018, 46, 22–27. [CrossRef]
- [19] Cibula, D.; Pötter, R.; Planchamp, F.; Avall-Lundqvist, E.; Fischerova, D.; Meder, C.H.; Köhler, C.; Landoni, F.; Lax, S.; Lindegaard, J.C.; et al. The European society of Gynaecological Oncology/European society for radiotherapy and Oncology/European society of pathology guidelines for the management of patients with cervical cancer. *Int. J. Gynecol. Cancer* 2018, 28, 641–655. [CrossRef] [PubMed]
- [20] Shi, P.; Zhang, L.; Ye, N. Sferummetabolomic analysis of cervical cancer patients by gas chromatography-mass spectrometry. *Asian J. Chem.* 2015, 27, 547–551.
- [21] Nishat, M. M., Faisal, F., Hasan, T., Karim, M. F. B., Islam, Z., and Shagor, M. R. K., “An Investigative Approach to Employ Support Vector Classifier as a

- Potential Detector of Brain Cancer from MRI Dataset,” 2021 International Conference on Electronics, Communication & Information Technology (ICECIT), pp. 1-4, IEEE, 2021, doi: 10.1109/ICECIT54077.2021.9641168
- [22] Nishat, M. M., Faisal, F., Dip, R. R., Shikder, M. F., Ahsan, R., Asif, M. A. A. R., and Udoy, M. R., “Performance Investigation of Different Boosting Algorithms in Predicting Chronic Kidney Disease,” In 2020 2nd International Conference on Sustainable Technologies for Industry 4.0 (STI), pp. 1-5. IEEE, 2020.
- [23] Asif, M. A. A. R., Nishat, M. M., Faisal, F., Dip, R. R., Udoy, M. H., Shikder, M. F., and Ahsan, R., “Performance Evaluation and Comparative Analysis of Different Machine Learning Algorithms in Predicting Cardiovascular Disease,” *Engineering Letters*, 29 (2), pp. 731-741, 2021
- [24] W. Wu and H. Zhou, “Data-driven diagnosis of cervical cancer with support vector machine-based approaches,” *IEEE Access*, vol. 5, pp. 25189–25195, 2017, doi: 10.1109/ACCESS.2017.2763984
- [25] Y. E. Kurniawati, A. E. Permanasari and S. Fauziati, "Comparative study on data mining classification methods for cervical cancer prediction using pap smear results," 2016 1st International Conference on Biomedical Engineering (IBIOMED), Yogyakarta, 2016, pp. 1-5.
- [26] Priyanka K Malli , Dr. Suvarna Nandyal, “Machine learning Technique for detection of Cervical Cancer using k-NN and Artificial Neural Network”, *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)*, July- August 2017.
- [27] R. Vidya1\* and G. M. Nasira2, “Prediction of Cervical Cancer using Hybrid Induction Technique: A Solution for Human Hereditary Disease Patterns”, *Indian Journal of Science and Technology*, August 2016.
- [28] D. Kashyap et al., "Cervical cancer detection and classification using Independent Level sets and multi SVMs," 2016 39th International Conference on Telecommunications and Signal Processing (TSP), Vienna, 2016, pp. 523-528.
- [29] E. Njoroge, S. R. Alty, M. R. Gani and M. Alkatib, "Classification of Cervical Cancer Cells using FTIR Data," 2006 International Conference of the IEEE Engineering in Medicine and Biology Society, New York, NY, 2006, pp. 5338-5341.

- [30] Y. S. Muhammad Fazal Ijaz, Muhammad Attique, "Data-Driven Cervical Cancer Prediction Model with Outlier Detection and Over-Sampling Methods," *Sensors*, vol. 20, no. 10, pp. 1424–8220, 2020.
- [31] J. Hyeon, H. J. Choi, K. N. Lee and B. D. Lee, "Automating Papanicolaou Test Using Deep Convolutional Activation Feature," 2017 18th IEEE International Conference on Mobile Data Management (MDM), Daejeon, 2017, pp. 382-385.
- [32] K. Teeyapan, N. Theera-Umpun and S. Auephanwiriyaikul, "Application of support vector based methods for cervical cancer cell classification," 2015 IEEE International Conference on Control System, Computing and Engineering (ICCSCE), George Town, 2015, pp. 514-519.
- [33] Sobar, R. Machmud, and A. Wijaya, "Behavior Determinant Based Cervical Cancer Early Detection with Machine Learning Algorithm," *Adv. Sci. Lett.*, vol. 22, no. 10, pp. 3120–3123, Oct. 2016, doi: 10.1166/ASL.2016.7980.
- [34] "UCI Machine Learning Repository: Cervical Cancer Behavior Risk Data Set." <https://archive.ics.uci.edu/ml/datasets/Cervical+Cancer+Behavior+Risk>
- [35] "Confusion Matrix" ([https://en.wikipedia.org/wiki/Confusion\\_matrix/](https://en.wikipedia.org/wiki/Confusion_matrix/))
- [36] H. Dalianis, *Clinical Text Mining: Secondary Use of Electronic Patient Records* Springer Open, Cham Switzerland (2018), p. 47, 10.1007/978-3-319-78503-5
- [37] Duboue, Pablo. (2020). *The Art of Feature Engineering: Essentials for Machine Learning*. 10.1017/9781108671682.
- [38] A. (2020, October 5). 7 Feature Engineering Techniques in Machine Learning You Should Know. Analytics Vidhya. <https://www.analyticsvidhya.com/blog/2020/10/7-feature-engineering-techniques-machine-learning/>
- [39] [https://www.saedsayad.com/k\\_nearest\\_neighbors.htm/](https://www.saedsayad.com/k_nearest_neighbors.htm/)
- [40] [https://www.tutorialspoint.com/machine\\_learning\\_with\\_python/machine\\_learning\\_with\\_python\\_knn\\_algorithm\\_finding\\_nearest\\_neighbors.htm/](https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_knn_algorithm_finding_nearest_neighbors.htm/)
- [41] <https://www.javatpoint.com/machine-learning-random-forest-algorithm/>
- [42] [https://en.wikipedia.org/wiki/Support-vector\\_machine/](https://en.wikipedia.org/wiki/Support-vector_machine/)
- [43] <https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm/>
- [44] <https://en.wikipedia.org/wiki/AdaBoost/>



- [45] <https://www.mygreatlearning.com/blog/xgboost-algorithm/>
- [46] <https://www.programmersought.com/article/16143908973/>
- [47] <https://www.hackerearth.com/practice/machine-learning/machine-learningalgorithms/beginners-tutorial-on-xgboost-parameter-tuning-r/tutorial/>
- [48] <https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm>
- [49] [https://iq.opengenus.org/gaussiannaivebayes/#:~:text=Gaussian%20Naive%20Bayes%20supports%20continuous,\(independen %20dimensions\)%20between%20dimensions.](https://iq.opengenus.org/gaussiannaivebayes/#:~:text=Gaussian%20Naive%20Bayes%20supports%20continuous,(independen%20dimensions)%20between%20dimensions.)
- [50] <https://catboost.ai/>
- [51] <https://catboost.ai/en/docs/concepts/algorithm-main-stages>
- [52] [https://en.wikipedia.org/wiki/Multilayer\\_perceptron](https://en.wikipedia.org/wiki/Multilayer_perceptron)
- [53] <https://www.analyticsvidhya.com/blog/2021/09/gradient-boosting-algorithm-a-complete-guide-for-beginners/>
- [54] <https://machinelearningmastery.com/random-forest-ensembles-with-xgboost/>