VISVESVARAYA TECHNOLOGICAL UNIVERSITY

'JNANA SANGAMA' BELAGAVI-590018, KARNATAKA

MINI-PROJECT REPORT

ON

"CANCER DETECTION USING MACHINE LEARNING"

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE V SEMESTER, BE, MINI-PROJECT-BCS586

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE & ENGINEERING

Submitted By

1. **VARSHA P N** **[1CG22CS119]**
2. **TRISHA M** **[1CG22CS116]**
3. **SUSHMITHA M R** **[1CG22CS114]**
4. **SHIVANI S R** **[1CG22CS104]**

Under the guidance of:

Mr. Harish T A **MTech.,**
Asst. Prof., Dept. of CSE,
CIT, Gubbi, Tumakuru .

HOD:

Dr. Shantala C P,Ph.D
Prof & Head, Dept. of CSE,
CIT, Gubbi, Tumakuru.

C.I.T
Partnering in Academic Excellence

Channabasaveshwara Institute of Technology
(NAAC Accredited &ISO 9001:2015 Certified Institution)
NH 206 (B.H. Road), Gubbi, Tumakuru – 572 216. Karnataka.

(Affiliated to Visvesvaraya Technological University, Belagavi & Recognized by AICTE New Delhi)

2024-25

Channabasaveshwara Institute of Technology
(NAAC Accredited &ISO 9001:2015 Certified Institution)
NH 206 (B.H. Road), Gubbi, Tumakuru – 572 216. Karnataka
(Affiliated to Visvesvaraya Technological University, Belagavi & Recognized by AICTE New Delhi)
2024-25

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the mini project work entitled **"CANCER DETECTION USING MACHINE LEARNING"** has been successfully carried out by **VARSHA P N[1CG22CS119] , TRISHA M[1CG22CS116] , SUSHMITHA M R[1CG22CS114] , SHIVANI S R[1CG22CS104]** bonafide students of **CHANNABASAVESHWARA INSTITUTE OF TECHNOLOGY, GUBBI, TUMAKURU,** under our supervision and guidance and submitted in partial fulfillment for V Semester BE, Mini-project-BCS586 by **Visvesvaraya Technological University, Belagavi** during the academic year of 2024–2025. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements for the above said degree.

Guide:

Mr. Harish T A **M.Tech,**
Asst. Prof., Dept. of CSE,
CIT, Gubbi, Tumakuru.

H.O.D:

Dr. Shantala C P **Ph.D**
Prof & Head,Dept. of CSE,
CIT, Gubbi, Tumakuru.

Principal:

Dr. Suresh D S **Ph.D**
Director & Principal
CIT, Gubbi, Tumakuru.

ABSTRACT

This study explores the development and evaluation of predictive models for Cancer detection using a dataset sourced from Kaggle. Cancer is one of the leading causes of death worldwide . This project focuses on leveraging machine learning techniques to predict cancer diagnoses based on clinical and diagnostic data. The goal is to build a model that can effectively classify cancer cases into malignant or benign categories .The dataset used in this study contains various features, such as the size, shape, and texture of tumors, along with other biological markers that can serve as indicators of cancer. Exploratory data analysis (EDA) and Multiple machine learning algorithms are applied to the task, including Support Vector Machine (SVM), Random Forest, and XGBoost, with hyperparameter tuning via GridSearchCV to optimize model performance. Models are evaluated using several metrics such as accuracy, precision, recall, F1-score, and confusion matrices, providing a comprehensive understanding of the classifier's performance. This machine learning-based approach supports medical professionals in detecting cancer more efficiently and accurately, ultimately contributing to better patient outcomes and improved healthcare strategies.

ACKNOWLEDGEMENT

A great deal of time and lot of effort has gone into completing this mini project report and documenting it. The number of hours spent in getting through various books and other materials related to this topic chosen by us have reaffirmed its power and utility in doing this work.

Several special people have contributed significantly to this effort. First of all, we are grateful to our institution **Channabasaveshwara Institute of Technology**, Gubbi, which provides us an opportunity in fulfilling our most cherished desire of reaching the goal.

We acknowledge and express our sincere thanks to our beloved Director and Principal, **Dr. Suresh D S** for his many valuable suggestions and continued encouragement and support in the academic endeavors.

We wish to express our deep sense of gratitude to **Dr**. **Shantala C P,** Prof & Head, Dept of CSE, CIT, for all the guidance and who still remains a constant driving force and motivated through innovative ideas with tireless support and advice during the course of mini project to examine and helpful suggestions offered.

We express our heartful thanks to our guide **Mr. Harish Athreya,** Assistant Professor, Dept. of CSE, CIT, for his meticulous attention to details, which has contributed immeasurably to the quality of our project report.

Our sincere and hearty thanks to our beloved parents, friends and others for their all-time support and co-operation.

PROJECT ASSOCIATES:

VARSHA P N [1CG22CS119]

TRISHA M [1CG22CS116]

SUSHMITHA M R [1CG22CS114]

SHIVANI S R [1CG22CS104]

Table of Contents

CHAPTER 1

INTRODUCTION

The integration of artificial intelligence and machine learning into healthcare is revolutionizing the way diseases are diagnosed and managed. This project focuses on building a robust machine learning model for cancer detection, leveraging advanced data analysis and detective modeling techniques. The goal is to classify tumors as either **malignant** (cancerous) or **benign** (non-cancerous) using data derived from fine needle aspirate images . By analyzing and interpreting features such as the mean radius, texture, perimeter, and compactness of cells, the project aims to assist in the early and accurate diagnosis of cancer. This early detection is crucial as it significantly increases the chances of successful treatment and survival.

The dataset used in this project is highly detailed and includes both numerical and categorical features representing the physical characteristics of the cells. However, one challenge lies in the imbalanced distribution of the target classes, with benign cases being more prevalent than malignant ones. To address this, the Synthetic Minority Oversampling Technique (SMOTE) is applied to balance the dataset, ensuring that the machine learning models are trained equitably and are sensitive to both classes.

The project employs a systematic workflow that begins with data preprocessing and exploratory analysis to identify key patterns and correlations. Feature selection is conducted using the Gini Importance measure from a Random Forest model, which helps in narrowing down the most impactful detectors. Multiple machine learning algorithms, including Random Forest, Support Vector Machines (SVM), and XGBoost, are trained and optimized using cross-validation and hyperparameter tuning via GridSearchCV to ensure the best possible performance.

This project underscores the potential of machine learning in revolutionizing healthcare diagnostics. By providing a fast, reliable, and automated tool for cancer classification, it complements traditional diagnostic methods, helping medical practitioners make informed decisions with greater confidence.

1.1 Objectives

- **Data Analysis :**Explore and understand the dataset, identifying key features contributing to the classification of tumors.
- **Feature Engineering:** Select the most important features to optimize model performance and reduce computational overhead.
- **Model Training:** Train multiple machine learning algorithms (Random Forest, SVM, and XGBoost) to classify tumors effectively.
- **Model Deployment:** Save and load the trained models for future predictions and deployment in real-world applications.

1.2 Problem Statement

Cancer, a leading global cause of mortality, involves abnormal cell growth that can spread and threaten life. Early and accurate detection is crucial for improving survival rates, but traditional methods like imaging and biopsies are often time-consuming, costly, and subject to human error. The challenge is to develop efficient, reliable, and scalable systems capable of analyzing complex biological data to detect cancer accurately. This project aims to utilize machine learning to build predictive models that assist in early diagnosis, reduce diagnostic delays, and support medical professionals in making precise and timely treatment decisions, ultimately enhancing patient outcomes.

CHAPTER 2

LITERATURE SURVEY

Machine learning (ML) has become a transformative tool in medical diagnostics, particularly for cancer detection. By analyzing complex and high-dimensional datasets, ML algorithms can uncover patterns and relationships that may be challenging for human experts to identify. This capability makes ML essential for tasks like classifying tumors as malignant or benign, providing timely and accurate results that can significantly improve patient outcomes.

Effective feature selection plays a critical role in cancer detection projects. High-dimensional medical datasets often include redundant or irrelevant features that can negatively affect model performance and interpretability. Studies demonstrate that techniques like Gini Importance in Random Forests or domain knowledge-driven feature prioritization are vital in isolating significant predictors such as tumor size, shape, or texture, thus enhancing the accuracy and efficiency of predictive models.

MACHINE LEARNING ALGORITHMS USED:

1. Random Forest Classifier

Random Forest, an ensemble learning method, is used for its robustness in handling complex data structures and its ability to manage overfitting through bootstrap aggregation. It evaluates the importance of each feature using Gini Importance, which helps identify the most significant factors for accurate predictions. This algorithm is highly effective in high-dimensional datasets like those in cancer detection.

2. Support Vector Machines (SVM)

SVM is a popular algorithm for classification tasks due to its ability to work well in high-dimensional spaces. By using kernel functions, SVMs can handle non-linear relationships between features. In this project, hyperparameters like C (regularization) and gamma (kernel coefficient) are optimized through GridSearchCV to maximize performance.

3. XGBoost (Extreme Gradient Boosting)

XGBoost is a gradient boosting algorithm known for its speed and performance. It iteratively improves model predictions by minimizing errors, making it particularly useful for structured datasets. In this project, XGBoost is fine-tuned using parameters like n_estimators to achieve high accuracy and reliability.

4. Synthetic Minority Oversampling Technique (SMOTE)

To address class imbalance in the dataset, SMOTE is employed to create synthetic examples of the minority class. This technique ensures the model learns effectively from both classes, improving its ability to identify malignant cases accurately.

These algorithms, combined with hyperparameter tuning and feature selection, form the foundation of the project, enabling robust and accurate cancer detection.

CHAPTER 3

SYSTEM DESIGN

3.1 ARCHITECTURE

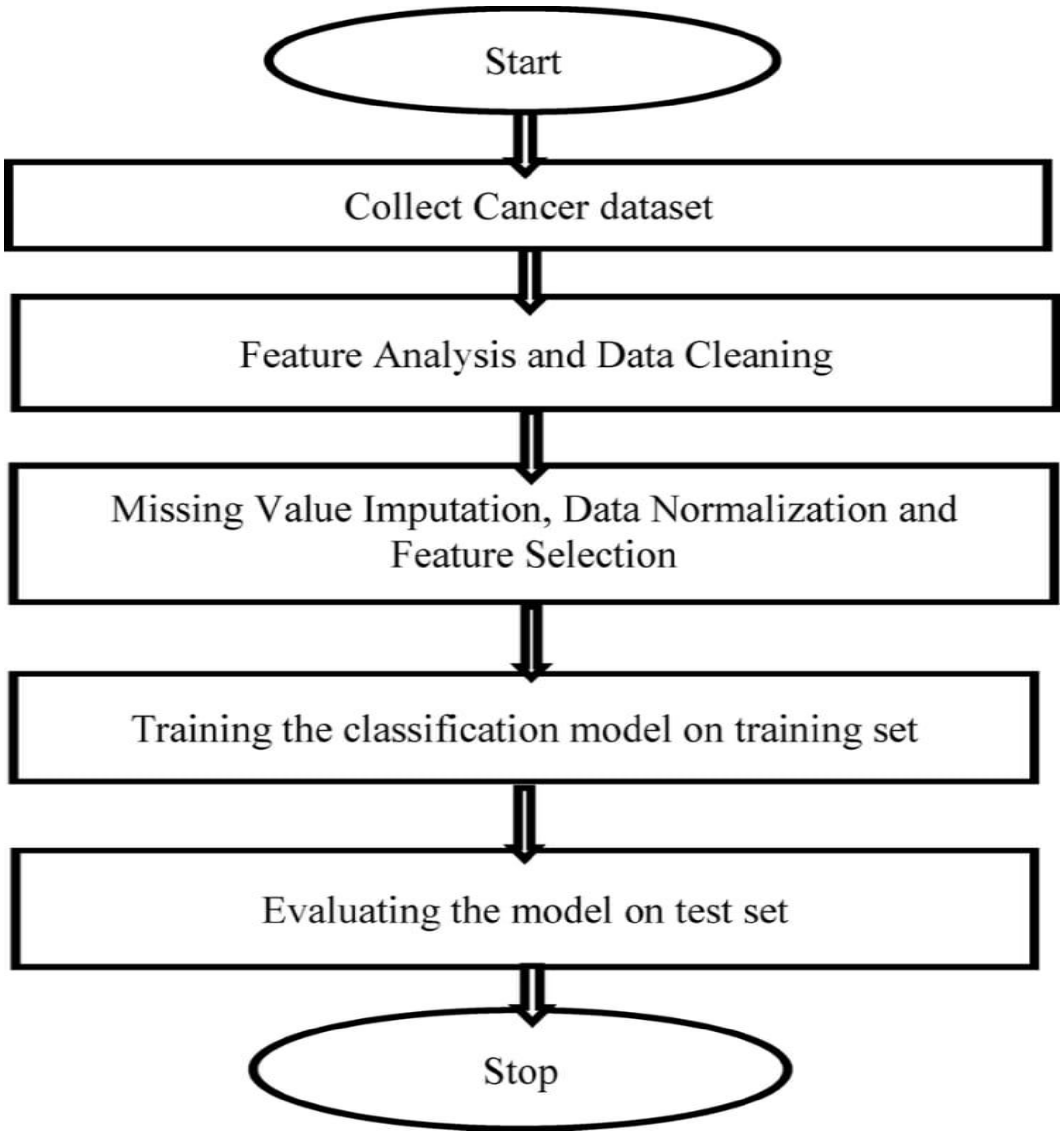

Start
Collect Cancer dataset
Feature Analysis and Data Cleaning
Missing Value Imputation, Data Normalization and Feature Selection
Training the classification model on training set
Evaluating the model on test set
Stop

Fig 3.1. Architecture of the proposed model

3.2 DATAFLOW DIAGRAM

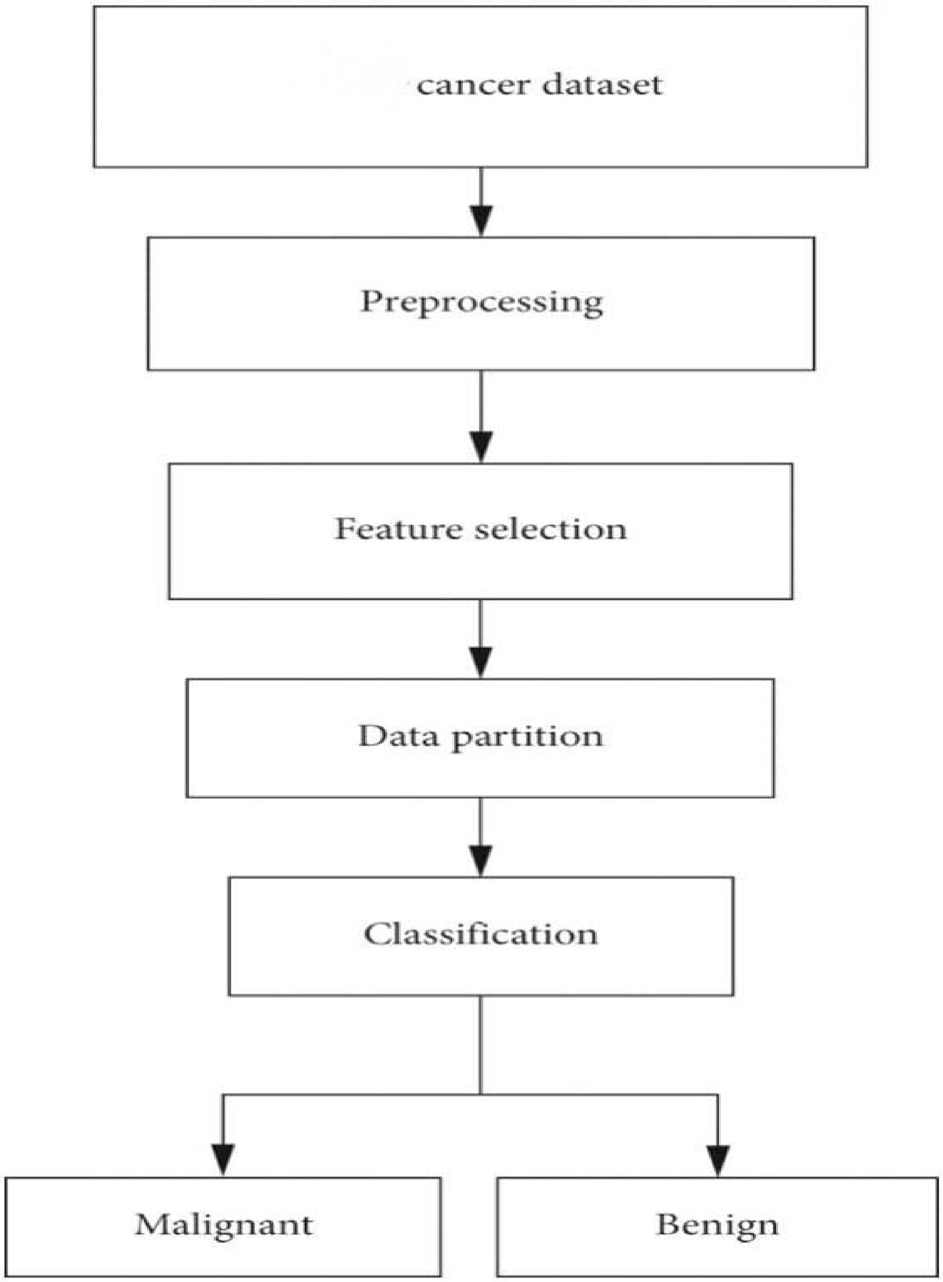


Fig 3.2. Dataflow diagram of proposed system

3.3 USE CASE DIAGRAM

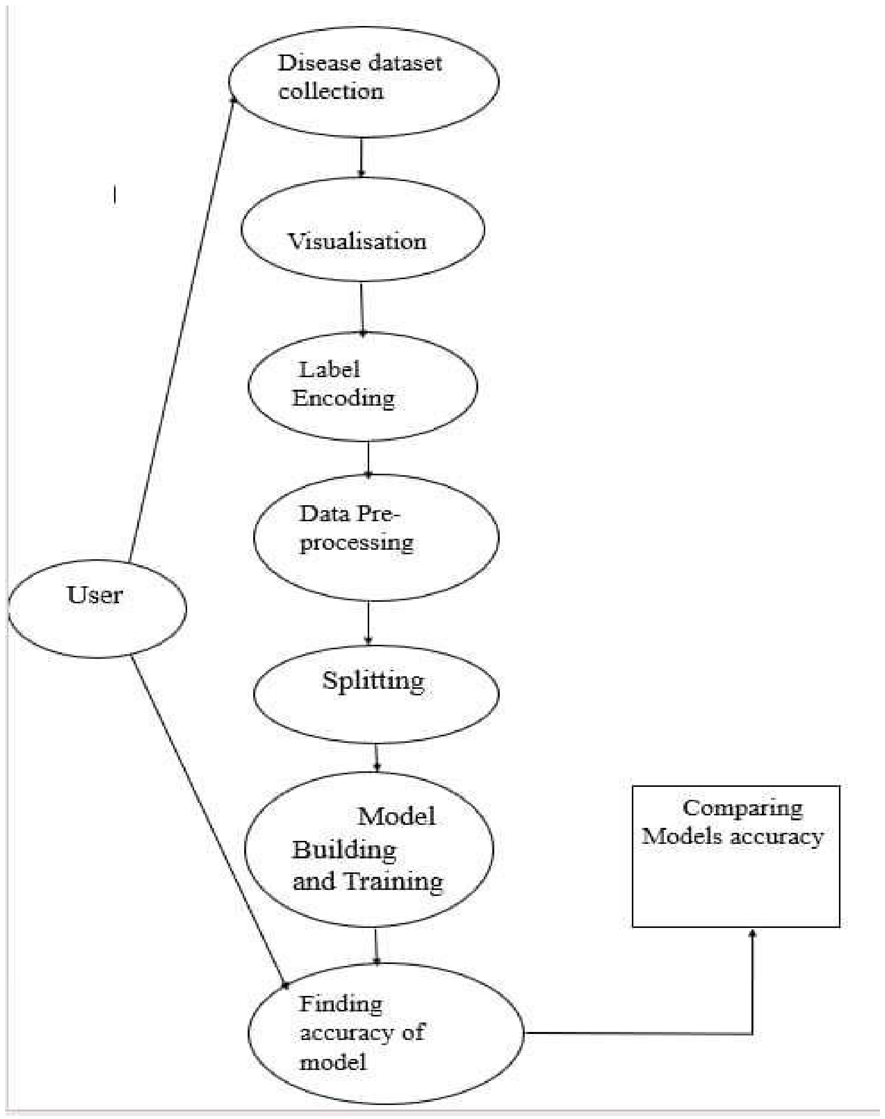
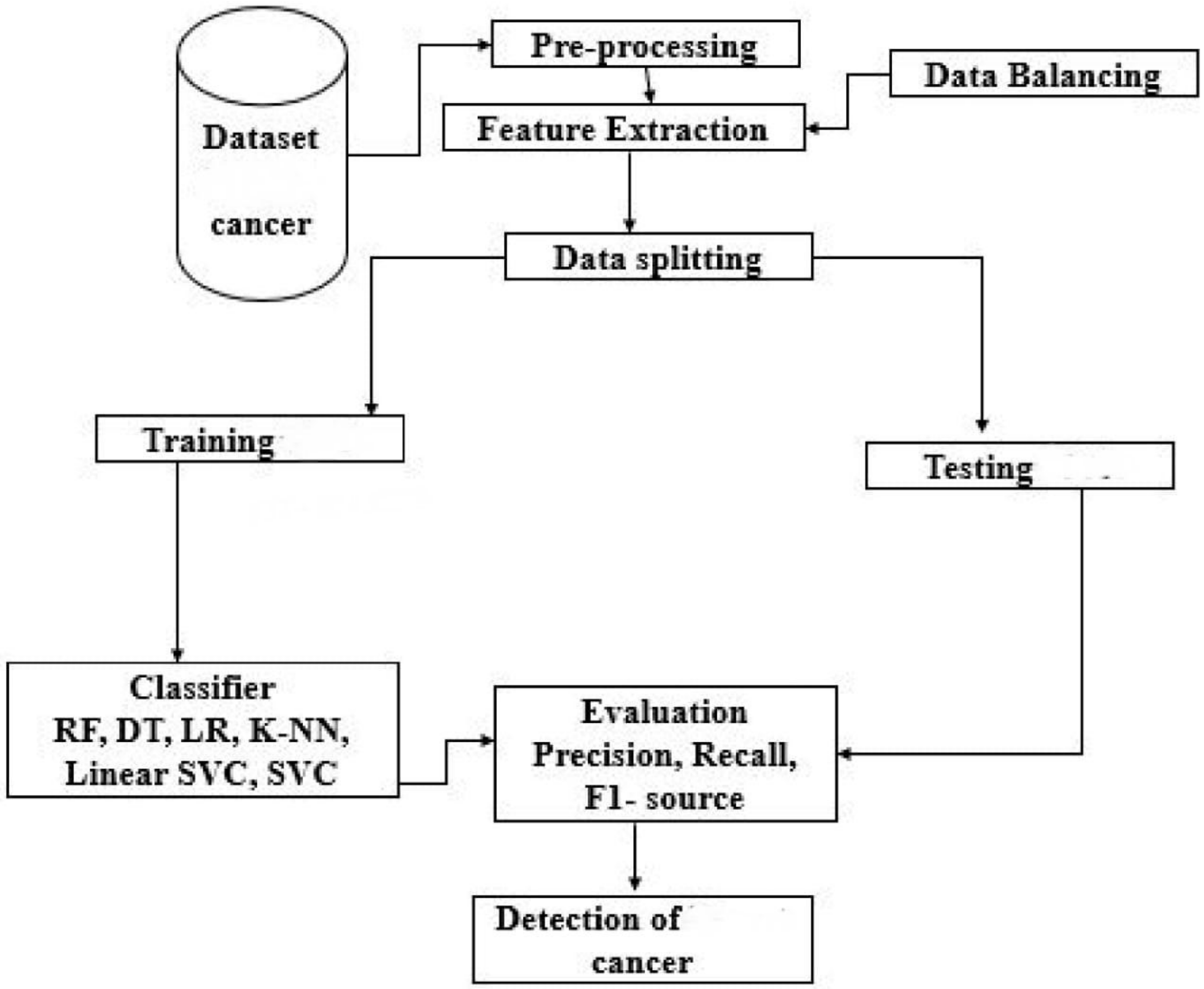

Disease dataset collection
Visualisation
Label Encoding
Data Pre-processing
User
Splitting
Model Building and Training
Comparing Models accuracy
Finding accuracy of model

Fig 3.3 use case diagram of proposed system

3.4 SEQUENCE DIAGRAM



3.4 Sequence diagram of proposed system

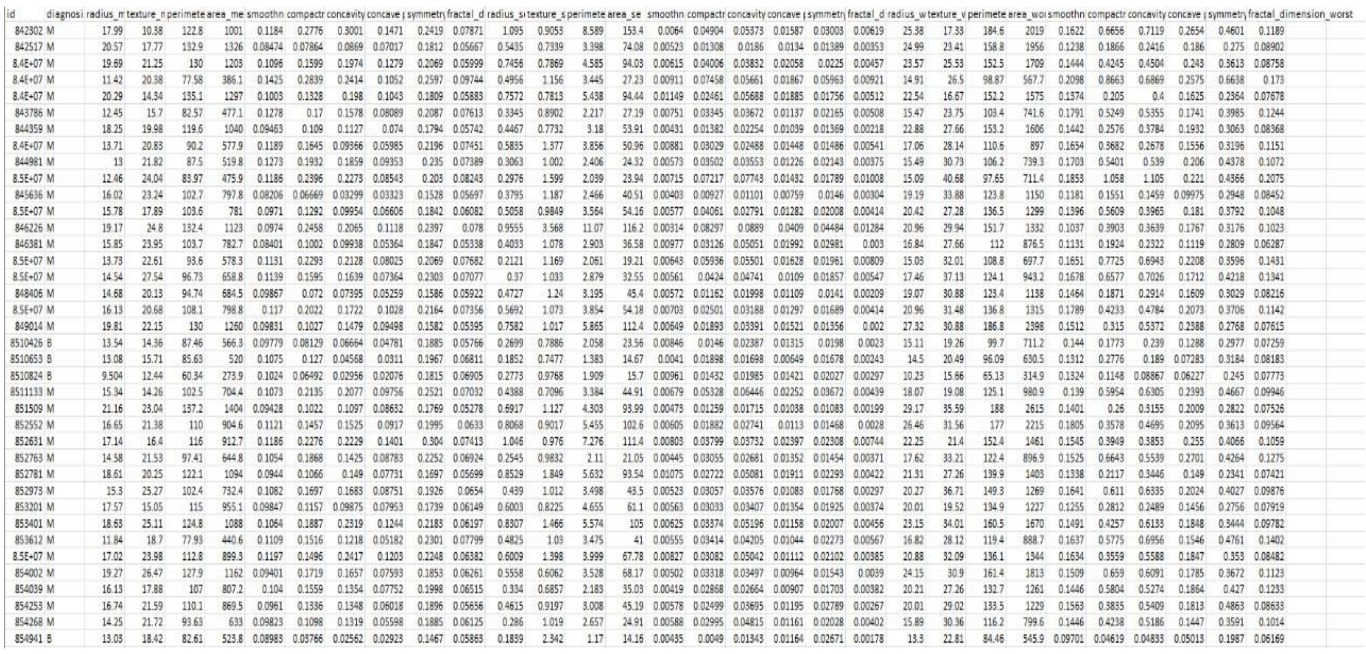
CHAPTER 4

METHODOLOGY

Methodology includes steps ranging from data acquisition and preprocessing to model evaluation and interpretation of results.

4.1 Data Collection

The dataset is downloaded from Kaggle. The Cancer Detection Model1.pynb file contains Cancer Data.csv file. The cancer dataset consists of 569 rows and 32 columns and Cancer data consist of 569 rows and 30 columns including independent column . and 1 column which is target(dependent)column(diagnosis).



id	diagnosi	radius_m	texture_m	perimete	area_me	smoothn	compactr	concavity	concave (	symmetr	fractal_d	radius_s	texture_s	perimete	area_se	smoothn	compactr	concavity	concave (	symmetr	fractal_d	radius_w	texture_v	perimete	area_woi	smoothn	compactr	concavity	concave (	symmetr	fractal_dimension_worst
842302	M	17.99	10.38	122.8	1001	0.1184	0.2776	0.3001	0.1471	0.2419	0.07871	1.095	0.9053	8.589	153.4	0.0064	0.04904	0.05373	0.01587	0.03003	0.00619	25.38	17.33	184.6	2019	0.1622	0.6656	0.7119	0.2654	0.4601	0.1189
842517	M	20.57	17.77	132.9	1326	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	0.5435	0.7339	3.398	74.08	0.00523	0.01308	0.0186	0.0134	0.01389	0.00353	24.99	23.41	158.8	1956	0.1238	0.1866	0.2416	0.186	0.275	0.08902
8.4E+07	M	19.69	21.25	130	1203	0.1096	0.1599	0.1974	0.1279	0.2069	0.05999	0.7456	0.7869	4.585	94.03	0.00615	0.04006	0.03832	0.02058	0.0225	0.00457	23.57	25.53	152.5	1709	0.1444	0.4245	0.4504	0.243	0.3613	0.08758
8.4E+07	M	11.42	20.38	77.58	386.1	0.1425	0.2839	0.2414	0.1052	0.2597	0.09744	0.4956	1.156	3.445	27.23	0.00911	0.07458	0.05661	0.01867	0.05963	0.00921	14.91	26.5	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.173
8.4E+07	M	20.29	14.34	135.1	1297	0.1003	0.1328	0.198	0.1043	0.1809	0.05883	0.7572	0.7813	5.438	94.44	0.01149	0.02461	0.05688	0.01885	0.01756	0.00512	22.54	16.67	152.2	1575	0.1374	0.205	0.4	0.1625	0.2364	0.07678
843786	M	12.45	15.7	82.57	477.1	0.1278	0.17	0.1578	0.08089	0.2087	0.07613	0.3345	0.8902	2.217	27.19	0.00751	0.03345	0.03672	0.01137	0.02165	0.00508	15.47	23.75	103.4	741.6	0.1791	0.5249	0.5355	0.1741	0.3985	0.1244
844359	M	18.25	19.98	119.6	1040	0.09463	0.109	0.1127	0.074	0.1794	0.05742	0.4467	0.7732	3.18	53.91	0.00431	0.01382	0.02254	0.01039	0.01369	0.00218	22.88	27.66	153.2	1606	0.1442	0.2576	0.3784	0.1932	0.3063	0.08368
8.4E+07	M	13.71	20.83	90.2	577.9	0.1189	0.1645	0.09366	0.05985	0.2196	0.07451	0.5835	1.377	3.856	50.96	0.00881	0.03029	0.02488	0.01448	0.01486	0.00541	17.06	28.14	110.6	897	0.1654	0.3682	0.2678	0.1556	0.3196	0.1151
844981	M	13	21.82	87.5	519.8	0.1273	0.1932	0.1859	0.09353	0.235	0.07389	0.3063	1.002	2.406	24.32	0.00573	0.03502	0.03553	0.01226	0.02143	0.00375	15.49	30.73	106.2	739.3	0.1703	0.5401	0.539	0.206	0.4378	0.1072
8.5E+07	M	12.46	24.04	83.97	475.9	0.1186	0.2396	0.2273	0.08543	0.203	0.08243	0.2976	1.599	2.039	23.94	0.00715	0.07217	0.07743	0.01432	0.01789	0.01008	15.09	40.68	97.65	711.4	0.1853	1.058	1.105	0.221	0.4366	0.2075
845636	M	16.02	23.24	102.7	797.8	0.08206	0.06669	0.03299	0.03323	0.1528	0.05697	0.3795	1.187	2.466	40.51	0.00403	0.00927	0.01101	0.00759	0.0146	0.00304	19.19	33.88	123.8	1150	0.1181	0.1551	0.1459	0.09975	0.2948	0.08452
8.5E+07	M	15.78	17.89	103.6	781	0.0971	0.1292	0.09954	0.06606	0.1842	0.06082	0.5058	0.9849	3.564	54.16	0.00577	0.04061	0.02791	0.01282	0.02008	0.00414	20.42	27.28	136.5	1299	0.1396	0.5609	0.3965	0.181	0.3792	0.1048
846226	M	19.17	24.8	132.4	1123	0.0974	0.2458	0.2065	0.1118	0.2397	0.078	0.9555	3.568	11.07	116.2	0.00314	0.08297	0.0889	0.0409	0.04484	0.01284	20.96	29.94	151.7	1332	0.1037	0.3903	0.3639	0.1767	0.3176	0.1023
846381	M	15.85	23.95	103.7	782.7	0.08401	0.1002	0.09938	0.05364	0.1847	0.05338	0.4033	1.078	2.903	36.58	0.00977	0.03126	0.05051	0.01992	0.02981	0.003	16.84	27.66	112	876.5	0.1131	0.1924	0.2322	0.1119	0.2809	0.06287
8.5E+07	M	13.73	22.61	93.6	578.3	0.1131	0.2293	0.2128	0.08025	0.2069	0.07682	0.2121	1.169	2.061	19.21	0.00643	0.05936	0.05501	0.01628	0.01961	0.00809	15.03	32.01	108.8	697.7	0.1651	0.7725	0.6943	0.2208	0.3596	0.1431
8.5E+07	M	14.54	27.54	96.73	658.8	0.1139	0.1595	0.1639	0.07364	0.2303	0.07077	0.37	1.033	2.879	32.55	0.00561	0.0424	0.04741	0.0109	0.01857	0.00547	17.46	37.13	124.1	943.2	0.1678	0.6577	0.7026	0.1712	0.4218	0.1341
848406	M	14.68	20.13	94.74	684.5	0.09867	0.072	0.07395	0.05259	0.1586	0.05922	0.4727	1.24	3.195	45.4	0.00572	0.01162	0.01998	0.01109	0.0141	0.00209	19.07	30.88	123.4	1138	0.1464	0.1871	0.2914	0.1609	0.3029	0.08216
8.5E+07	M	16.13	20.68	108.1	798.8	0.117	0.2022	0.1722	0.1028	0.2164	0.07356	0.5692	1.073	3.854	54.18	0.00703	0.02501	0.03188	0.01297	0.01689	0.00414	20.96	31.48	136.8	1315	0.1789	0.4233	0.4784	0.2073	0.3706	0.1142
849014	M	19.81	22.15	130	1260	0.09831	0.1027	0.1479	0.09498	0.1582	0.05395	0.7582	1.017	5.865	112.4	0.00649	0.01893	0.03391	0.01521	0.01356	0.002	27.32	30.88	186.8	2398	0.1512	0.315	0.5372	0.2388	0.2768	0.07615
8510426	B	13.54	14.36	87.46	566.3	0.09779	0.08129	0.06664	0.04781	0.1885	0.05766	0.2699	0.7886	2.058	23.56	0.00846	0.0146	0.02387	0.01315	0.0198	0.0023	15.11	19.26	99.7	711.2	0.144	0.1773	0.239	0.1288	0.2977	0.07259
8510653	B	13.08	15.71	85.63	520	0.1075	0.127	0.04568	0.0311	0.1967	0.06811	0.1852	0.7477	1.383	14.67	0.0041	0.01898	0.01698	0.00649	0.01678	0.00243	14.5	20.49	96.09	630.5	0.1312	0.2776	0.189	0.07283	0.3184	0.08183
8510824	B	9.504	12.44	60.34	273.9	0.1024	0.06492	0.02956	0.02076	0.1815	0.06905	0.2773	0.9768	1.909	15.7	0.00961	0.01432	0.01985	0.01421	0.02027	0.00297	10.23	15.66	65.13	314.9	0.1324	0.1148	0.08867	0.06227	0.245	0.07773
8511133	M	15.34	14.26	102.5	704.4	0.1073	0.2135	0.2077	0.09756	0.2521	0.07032	0.4388	0.7096	3.384	44.91	0.00679	0.05328	0.06446	0.02252	0.03672	0.00439	18.07	19.08	125.1	980.9	0.139	0.5954	0.6305	0.2393	0.4667	0.09946
851509	M	21.16	23.04	137.2	1404	0.09428	0.1022	0.1097	0.08632	0.1769	0.05278	0.6917	1.127	4.303	93.99	0.00473	0.01259	0.01715	0.01038	0.01083	0.00199	29.17	35.59	188	2615	0.1401	0.26	0.3155	0.2009	0.2822	0.07526
852552	M	16.65	21.38	110	904.6	0.1121	0.1457	0.1525	0.0917	0.1995	0.0633	0.8068	0.9017	5.455	102.6	0.00605	0.01882	0.02741	0.0113	0.01468	0.0028	26.46	31.56	177	2215	0.1805	0.3578	0.4695	0.2095	0.3613	0.09564
852631	M	17.14	16.4	116	912.7	0.1186	0.2276	0.2229	0.1401	0.304	0.07413	1.046	0.976	7.276	111.4	0.00803	0.03799	0.03732	0.02397	0.02308	0.00744	22.25	21.4	152.4	1461	0.1545	0.3949	0.3853	0.255	0.4066	0.1059
852763	M	14.58	21.53	97.41	644.8	0.1054	0.1868	0.1425	0.08783	0.2252	0.06924	0.2545	0.9832	2.11	21.05	0.00445	0.03055	0.02681	0.01352	0.01454	0.00371	17.62	33.21	122.4	896.9	0.1525	0.6643	0.5539	0.2701	0.4264	0.1275
852781	M	18.61	20.25	122.1	1094	0.0944	0.1066	0.149	0.07731	0.1697	0.05699	0.8529	1.849	5.632	93.54	0.01075	0.02722	0.05081	0.01911	0.02293	0.00422	21.31	27.26	139.9	1403	0.1338	0.2117	0.3446	0.149	0.2341	0.07421
852973	M	15.3	25.27	102.4	732.4	0.1082	0.1697	0.1683	0.08751	0.1926	0.0654	0.439	1.012	3.498	43.5	0.00523	0.03057	0.03576	0.01083	0.01768	0.00297	20.27	36.71	149.3	1269	0.1641	0.611	0.6335	0.2024	0.4027	0.09876
853201	M	17.57	15.05	115	955.1	0.09847	0.1157	0.09875	0.07953	0.1739	0.06149	0.6003	0.8225	4.655	61.1	0.00563	0.03033	0.03407	0.01354	0.01925	0.00374	20.01	19.52	134.9	1227	0.1255	0.2812	0.2489	0.1456	0.2756	0.07919
853401	M	18.63	25.11	124.8	1088	0.1064	0.1887	0.2319	0.1244	0.2183	0.06197	0.8307	1.466	5.574	105	0.00625	0.03374	0.05196	0.01158	0.02007	0.00456	23.15	34.01	160.5	1670	0.1491	0.4257	0.6133	0.1848	0.3444	0.09782
853612	M	11.84	18.7	77.93	440.6	0.1109	0.1516	0.1218	0.05182	0.2301	0.07799	0.4825	1.03	3.475	41	0.00555	0.03414	0.04205	0.01044	0.02273	0.00567	16.82	28.12	119.4	888.7	0.1637	0.5775	0.6956	0.1546	0.4761	0.1402
8.5E+07	M	17.02	23.98	112.8	899.3	0.1197	0.1496	0.2417	0.1203	0.2248	0.06382	0.6009	1.398	3.999	67.78	0.00827	0.03082	0.05042	0.01112	0.02102	0.00385	20.88	32.09	136.1	1344	0.1634	0.3559	0.5588	0.1847	0.353	0.08482
854002	M	19.27	26.47	127.9	1162	0.09401	0.1719	0.1657	0.07593	0.1853	0.06261	0.5558	0.6062	3.528	68.17	0.00502	0.03318	0.03497	0.00964	0.01543	0.0039	24.15	30.9	161.4	1813	0.1509	0.659	0.6091	0.1785	0.3672	0.1123
854039	M	16.13	17.88	107	807.2	0.104	0.1559	0.1354	0.07752	0.1998	0.06515	0.334	0.6857	2.183	35.03	0.00419	0.02868	0.02664	0.00907	0.01703	0.00382	20.21	27.26	132.7	1261	0.1446	0.5804	0.5274	0.1864	0.427	0.1233
854253	M	16.74	21.59	110.1	869.5	0.0961	0.1336	0.1348	0.06018	0.1896	0.05656	0.4615	0.9197	3.008	45.19	0.00578	0.02499	0.03695	0.01195	0.02789	0.00267	20.01	29.02	133.5	1229	0.1563	0.3835	0.5409	0.1813	0.4863	0.08633
854268	M	14.25	21.72	93.63	633	0.09823	0.1098	0.1319	0.05598	0.1885	0.06125	0.286	1.019	2.657	24.91	0.00588	0.02995	0.04815	0.01161	0.02028	0.00402	15.89	30.36	116.2	799.6	0.1446	0.4238	0.5186	0.1447	0.3591	0.1014
854941	B	13.03	18.42	82.61	523.8	0.08983	0.03766	0.02562	0.02923	0.1467	0.05863	0.1839	2.342	1.17	14.16	0.00435	0.0049	0.01343	0.01164	0.02671	0.00178	13.3	22.81	84.46	545.9	0.09701	0.04619	0.04833	0.05013	0.1987	0.06169

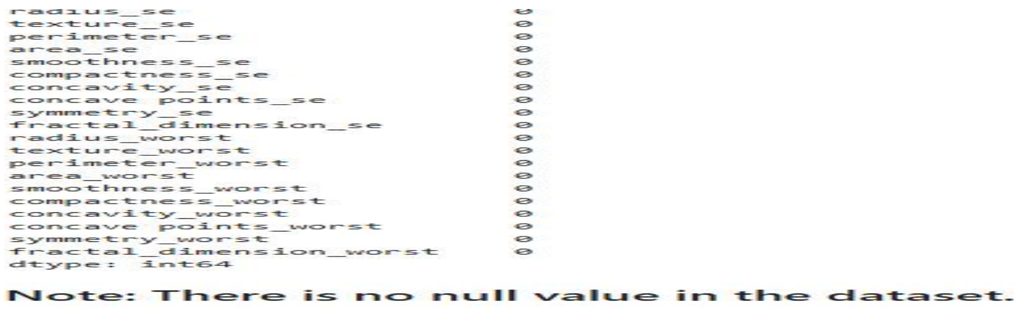
Fig.4.1.1 total 32columns in data set

4.2 Data Preprocessing:

In the data preprocessing step, the primary focus was to prepare the dataset for analysis and modelling. The dataset consisted of 32 columns, where 32 columns represent feature selection, and the target column (labelled diagnosis) detects the presence or absence of cancer. The independent features are float64 values. The dependent column represented by M and B, where:

- M indicates the presence of a Cancer.
- B indicates the absence of a Cancer.

➢ To ensure data quality, the .isnull () function from the Pandas library in Python was employed. This function was used to check for any missing (null) values across all columns in the dataset. Upon running the check, it was confirmed that no null values were present in any of the columns, ensuring a complete and clean dataset for further analysis.



```
radius_se                  0
texture_se                 0
perimeter_se               0
area_se                    0
smoothness_se              0
compactness_se             0
concavity_se               0
concave points_se          0
symmetry_se                0
fractal_dimension_se       0
radius_worst               0
texture_worst              0
perimeter_worst            0
area_worst                 0
smoothness_worst           0
compactness_worst          0
concavity_worst            0
concave points_worst       0
symmetry_worst             0
fractal_dimension_worst    0
dtype: int64
```

Note: There is no null value in the dataset.

Fig.4.2.1.Checking of null values in dataset

➢ Categorical Encoding: Convert categorical variables (like the diagnosis column with values 'M' and 'B') into numeric values (0 for benign and 1 for malignant) using Label Encoder or similar methods.

➢ Feature Scaling: Scale numeric features (such as radius, texture, area, etc.) to standardize the data for models that require it (e.g., SVM, k-NN).

4.3 Exploratory Data Analysis(EDA)

4.3.1 Visualizing Data

To gain deeper insights into the dataset and understand the relationships between features and the target column (diagnosis), various visualization techniques were employed:

1. **HeatMap plot**

   heatmap is a graphical representation of data that uses a system of color coding to represent different values. Heatmaps are used in various forms of analytics but are most commonly used to show relationships between two variables,one plotted on each axis and make it easier to discern areas of high and low concentration.

2. **Box Plot**

   A boxplot,is a standardized way of displaying the distribution of a data set based on its five-number summary of data points used to show distributions of numeric data values. it also organizes large amounts of data, and visualizes outlier values.

3. **Distribution Plot**

   A distribution plot is a visualization that shows the distribution of numerical data in a data set. It can be used to compare the range and distribution of data groups.Distribution plots compare observed data to expected outcomes. They display two plots for the same variable: a histogram that represents the observed data, and a density curve that represents the expected outcomes.

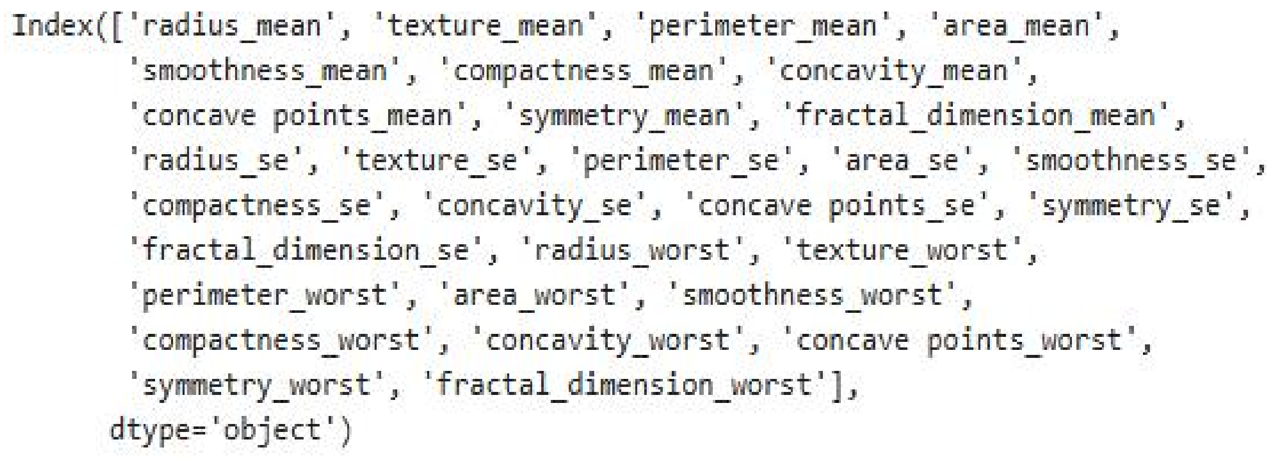
4.3.2 Class Imbalance Detection

Visualize the distribution of benign and malignant classes to check if the dataset is imbalanced. If one class dominates, techniques like SMOTE (Synthetic Minority Over-sampling Technique) can be used to balance the data.

4.4 Data Splitting:

1. Overview of Data Splitting:

- The dataset contains features that describe various characteristics of cell nuclei (such as radius_mean, texture_mean, area_mean, etc.) and a target variable (diagnosis) indicating whether the tumor is Malignant (M) or Benign (B).
- The purpose of data splitting is to divide the dataset into separate sets for training and testing to avoid overfitting and ensure the model generalizes well on unseen data.



```
Index(['radius_mean', 'texture_mean', 'perimeter_mean', 'area_mean',
       'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst'],
      dtype='object')
```

Fig.4.4.1 Features of dataset

2. Train-Test Split:

- Training Set: This subset of the data is used to train the machine learning models, allowing the model to learn from the features and their relationship to the target variable.
- Testing Set: This subset is kept aside during training and is used to evaluate the performance of the trained model. The testing data is not seen by the model during training, providing an unbiased estimate of the model's performance.
- Typically, 80% of the data is used for training, and 20% for testing. This is known as an 80-20 split, but in some cases, a different split ratio like 70-30 is used based on dataset size and model requirements.

3. Cross-Validation :

- Cross-validation is a technique used to ensure that the model performs well across different subsets of the data. It divides the dataset into multiple folds (e.g., 10-fold cross-validation).
- KFold Cross-Validation is used,data is split into K number of subsets and model is trained and tested K times, each time with a different fold as the test set and the remaining folds as the training set. This helps provide a more reliable performance estimate by averaging the results over different test sets.

4. Stratified Sampling :

- In case the dataset has an imbalanced class distribution(more Benign than Malignant cases), Stratified Sampling can be used. This ensures that both the training and testing sets have a proportionate representation of each class (Malignant and Benign) based on their occurrence in the full dataset.
- This prevents the model from being biased toward the majority class and helps it learn patterns for both Malignant and Benign cases.

5. Data Resampling :

- SMOTE (Synthetic Minority Over-sampling Technique): If the dataset is highly imbalanced (for example, if there are significantly more Benign cases than Malignant), SMOTE can be used to oversample the minority class (Malignant) in the training set by generating synthetic samples. This helps in balancing the dataset, allowing the model to perform better in detecting Malignant cases.

6. Splitting Code Implementation:

- The data is typically split using a function like train_test_split() from scikit-learn, which handles the random shuffling and splitting process. For cross-validation, KFold from scikit-learn can be used to generate train-test splits across different folds.

4.5 Feature Selection:

- feature selection is a crucial step to ensure the model uses the most informative attributes for accurate classification. Initially, the dataset contains several features related to the characteristics of cell nuclei, such as radius_mean, texture_mean, and area_mean. To identify the most relevant features, the project uses Random Forest's feature importance method. This technique ranks the features based on how much they contribute to reducing the Gini impurity during the tree-building process. The top-ranked features, like radius_mean and texture_mean, are then selected for model training, while less relevant or redundant features are discarded. This helps improve model performance, reduce computation time, and avoid overfitting.

4.6 Model used:

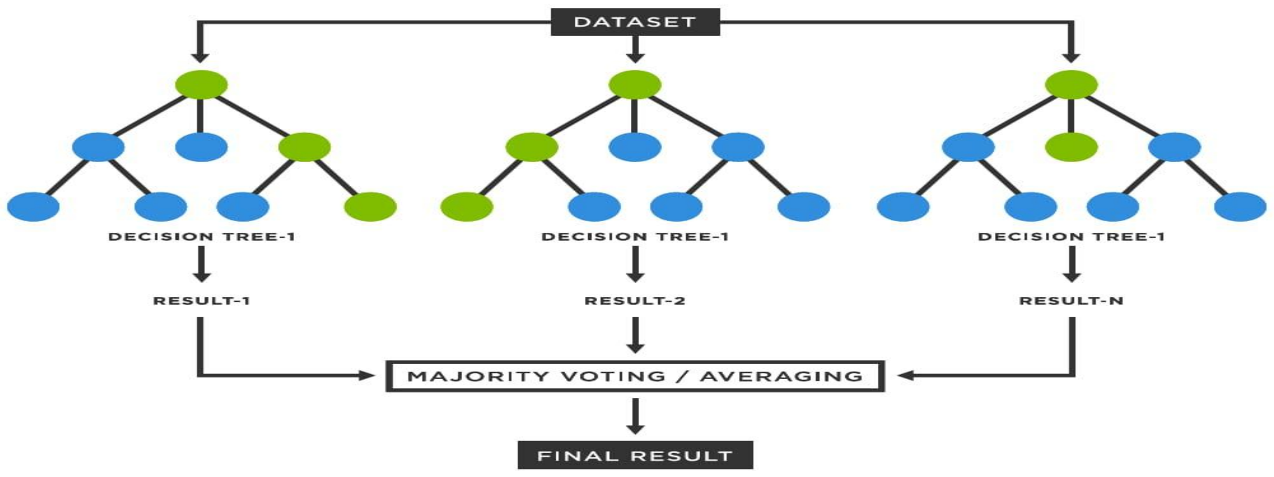
the project employs three machine learning models.

4.6.1 Random Forest Classifier

- Random Forest is a supervised machine learning algorithm extensively used for both classification and regression tasks. In the context of classification, such as diagnosing diseases, Random Forest excels at analyzing complex relationships in data and making robust predictions based on multiple input features like patient symptoms or test results.

- This algorithm operates by building an ensemble of decision trees, where each tree makes an independent prediction. The final classification is determined by aggregating the predictions through majority voting, enhancing the model's accuracy and reducing the risk of overfitting.

- A key feature of Random Forest is its use of randomness in both sampling the training data (bootstrap sampling) and selecting subsets of features for splitting nodes within each tree. This ensures diversity among the trees and leads to better



generalization to unseen data.

Fig 4.6.1 Random Forest

ALGORITHM

Step 1: Collect and Prepare Data

Gather and preprocess the dataset for classification. Handle missing values, remove duplicates, and encode categorical variables as needed. Split the dataset into training and testing sets.

Step 2: Initialize the Random Forest

Specify the hyper parameters, including the number of trees maximum tree depth, and other parameters like minimum sample ssplit. Note that the Random Forest automatically applies bootstrap sampling during training.

Step 3: Train the Model

Fit the Random Forest Classifier to the training data. The algorithm automatically:

- Builds multiple decision trees by sampling subsets of data (handled internally).
- Selects a random subset of features at each split within the decision trees.

Step 4: Evaluate the Model

Use the testing set to evaluate the model's performance. Predictions from individual trees are aggregated (majority voting for classification). Assess model accuracy and other metrics like precision, recall, F1- score.

Step 5: Make Predictions

Apply the trained Random Forest Classifier to new, unseen data for classification. Optionally, interpret feature importance scores to understand which features significantly influence predictions

4.6.2 Support Vector Machine

- Support Vector Machine (SVM) is a supervised machine learning algorithm commonly used for classification tasks, such as diagnosing diseases by distinguishing between different health conditions based on patient data like symptoms or test results.
- SVM aims to find the optimal hyperplane that separates data points into predefined classes with the maximum margin between the closest points (support vectors) of each class. This margin maximization ensures better generalization of the model to unseen data.
- In cases where data is not linearly separable, SVM uses kernel functions (e.g., linear, polynomial, radial basis function) to map the input data into a higher-dimensional space, making separation possible.

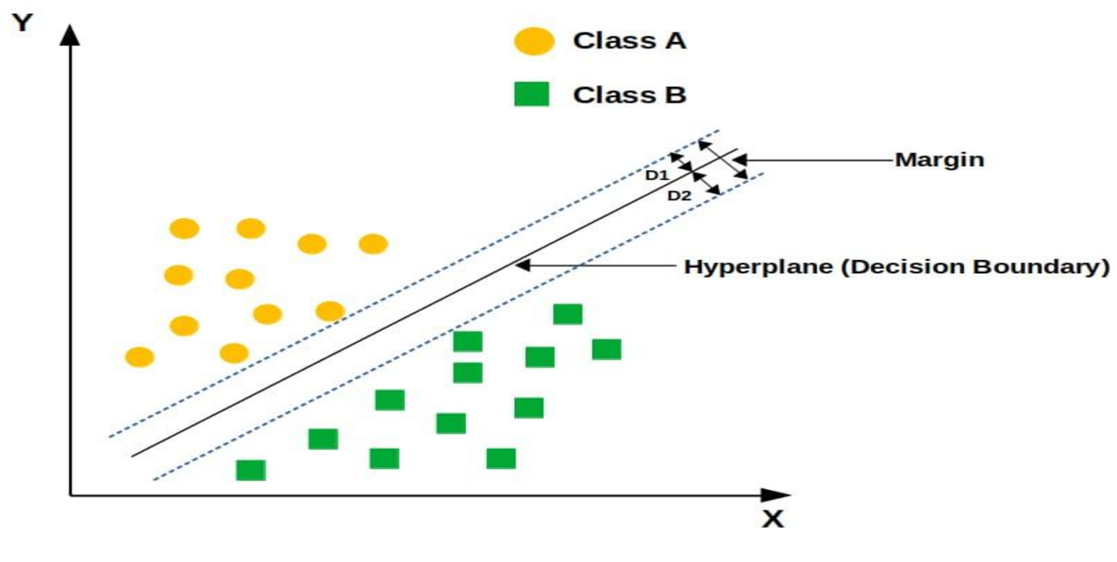


Fig 4.6.2 Support Vector Machine

ALGORITHM

Step 1: Collect and Prepare Data

Gather and clean the dataset for your classification task. Handle missing values, remove duplicates, and preprocess the data. Scale the features (e.g., standardization or normalization) because SVMs are sensitive to the range of feature values. Split the dataset into training and testing sets.

Step 2: Define the Model

Choose the SVM type based on the problem:

- Linear SVM for linearly separable data.
- Kernel SVM (e.g., RBF, polynomial, or sigmoid kernel) for non-linearly separable data. Specify hyper parameters such as C (regularization parameter) and kernel type.

Step 3: Train the Model

- Fit the SVM model to the training data. The algorithm:
- Finds the hyper plane that maximizes the margin (distance) between the two classes.
- Considers a small margin of error, controlled by the regularization parameter C, for misclassified points.
- For non-linear data, maps the input space to a higher-dimensional feature space using the selected kernel function.

Step 4: Evaluate the Model

Use the testing set to predict the target variable. Evaluate the model’s performance using metricslike accuracy, precision, recall, F1-score. Analyze the support vectors (data points closest to the hyper plane) to understand the classification boundaries.

Step 5: Make Predictions

Apply the trained SVM model to new, unseen data for classification. Interpret the decision boundaries and optionally perform hyper parameter tuning (e.g., grid search) to optimize the model for better performance.

4.6.3 XG Boost

- XG Boost is a high-performance, supervised machine learning algorithm based on the gradient boosting framework. It is widely used for both classification and regression tasks, including disease diagnosis, due to its speed and predictive power.
- XG Boost builds an ensemble of decision trees in a sequential manner, where each tree corrects the errors made by its predecessor. The final prediction is an aggregation of the individual trees' outputs, weighted by their learning contributions.
- What sets XG Boost apart is its focus on regularization (to prevent overfitting), parallel processing, and efficient handling of missing values, making it suitable for large datasets with complex feature interactions.

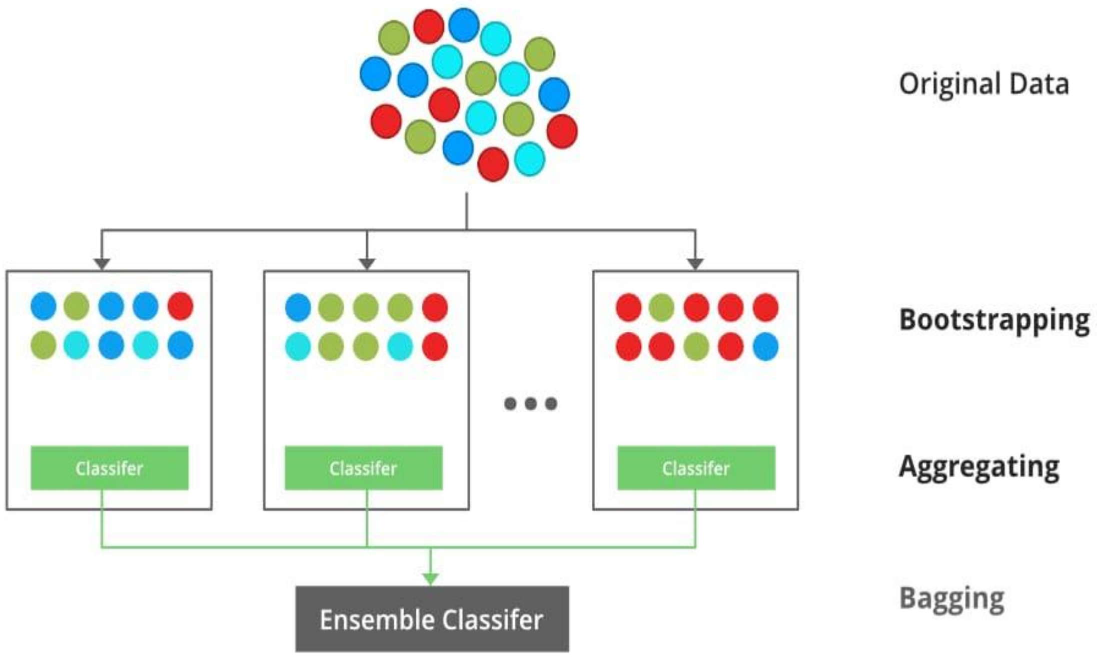


Fig 4.6.3 XG Boost

ALGORITHM

Step 1: Collect and Prepare Data

Gather and clean the dataset for the classification task. Handle missing values, remove duplicates, and preprocess the features (e.g., encoding categorical variables, scaling). Split the dataset into training and testing sets.

Step 2: Define the Model

Initialize the XG Boost model with hyper parameters such as:

Number of boosting rounds ,The step size at each iteration , Maximum depth of each decision tree, Fraction of training data used in each boosting round, Fraction of features used for each tree, Loss function.

Step 3: Train the Model

Fit the XG Boost model to the training data using gradient boosting. In each boosting round:

- Train a decision tree on the residual errors from the previous tree.
- Update the model by adding the new tree's predictions weighted by the learning rate.
- Repeat the process for the specified number of boosting rounds (or until convergence).

Step 4: Evaluate the Model

Use the testing set to evaluate the model’s performance. Predict the target variable and evaluate performance using metrics like accuracy, precision, recall, F1-score. Optionally, perform cross- validation for more robust results.

Step 5: Make Predictions

Apply the trained XG Boost model to new, unseen data for classification. Optionally, tune hyper parameters (using grid search or random search) to improve the model's performance.

4.7 Model Evaluation

Accuracy: This is the proportion of correct predictions out of all the predictions made by the model. However, accuracy can be misleading if the data is imbalanced ,as a model that always predicts the majority class can have a high accuracy but not be useful. Mathematically, accuracy is calculated as the ration of the number of correctly predicted instances to the total number of instances:

$$\textbf{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total Number of Predictions}}$$

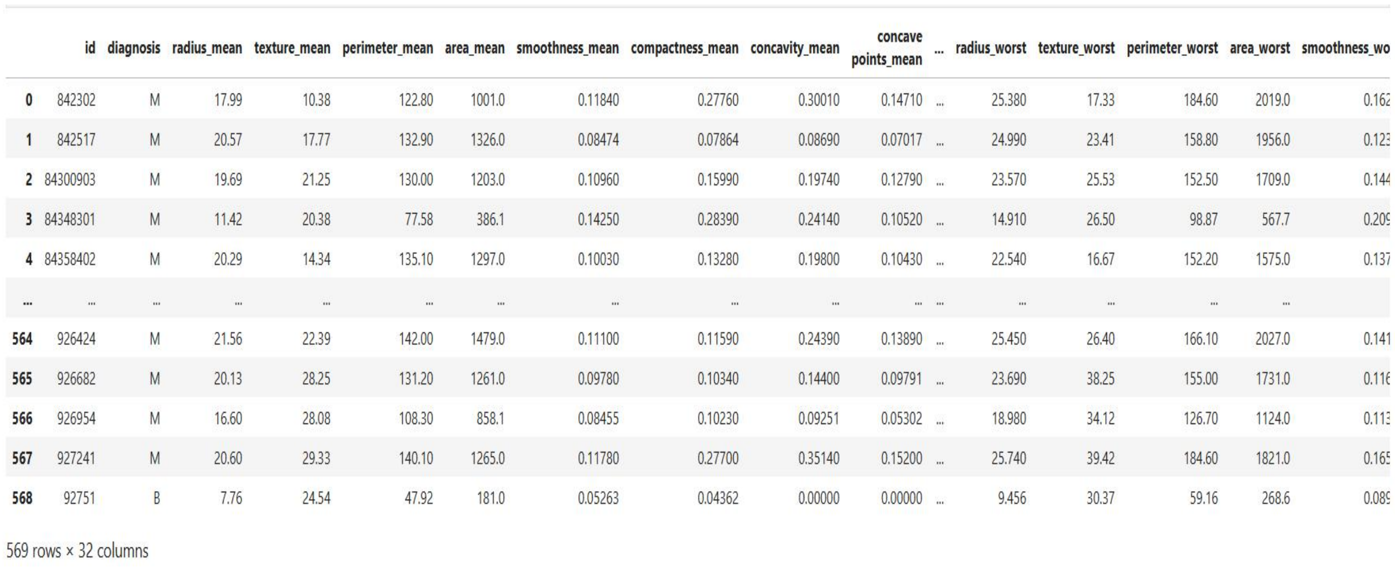
Recall:

- Recall, also known as sensitivity or true positive rate, measures the proportion of true positive predictions (correctly predicted positive instances) out of all actual positive instances in the dataset.
- It focuses on the model's ability to capture all positive instances and answers the question: "Of all actual positive instances, how many did the model correctly predict as positive?"

F1-score: This is a harmonic mean of precision and recall, and can be useful for imbalanceddatasets where both precision an drew al l need to be considered.

CHAPTER 5

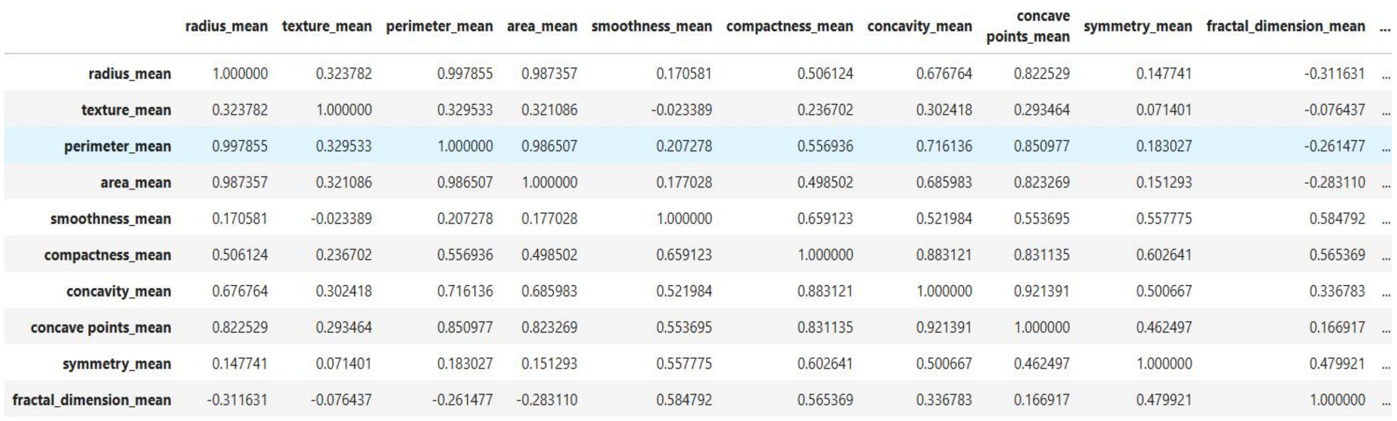
RESULTS



	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...	radius_worst	texture_worst	perimeter_worst	area_worst	smoothness_wo
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	...	25.380	17.33	184.60	2019.0	0.162
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	...	24.990	23.41	158.80	1956.0	0.123
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	...	23.570	25.53	152.50	1709.0	0.144
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	...	14.910	26.50	98.87	567.7	0.209
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	...	22.540	16.67	152.20	1575.0	0.137
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	...	25.450	26.40	166.10	2027.0	0.141
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	...	23.690	38.25	155.00	1731.0	0.116
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	...	18.980	34.12	126.70	1124.0	0.113
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	...	25.740	39.42	184.60	1821.0	0.165
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	...	9.456	30.37	59.16	268.6	0.089

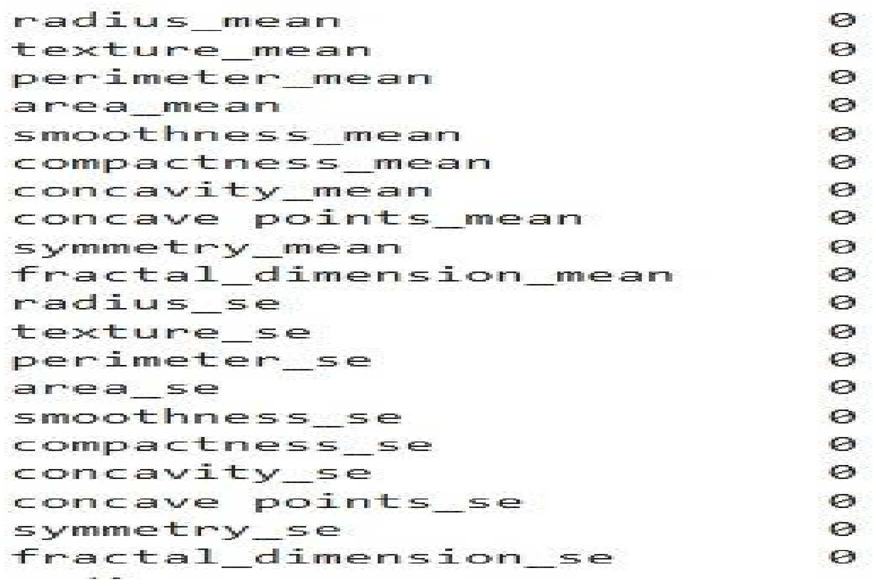
569 rows × 32 columns

Fig 5.1 Reading Data Set



	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symmetry_mean	fractal_dimension_mean	...
radius_mean	1.000000	0.323782	0.997855	0.987357	0.170581	0.506124	0.676764	0.822529	0.147741	-0.311631	...
texture_mean	0.323782	1.000000	0.329533	0.321086	-0.023389	0.236702	0.302418	0.293464	0.071401	-0.076437	...
perimeter_mean	0.997855	0.329533	1.000000	0.986507	0.207278	0.556936	0.716136	0.850977	0.183027	-0.261477	...
area_mean	0.987357	0.321086	0.986507	1.000000	0.177028	0.498502	0.685983	0.823269	0.151293	-0.283110	...
smoothness_mean	0.170581	-0.023389	0.207278	0.177028	1.000000	0.659123	0.521984	0.553695	0.557775	0.584792	...
compactness_mean	0.506124	0.236702	0.556936	0.498502	0.659123	1.000000	0.883121	0.831135	0.602641	0.565369	...
concavity_mean	0.676764	0.302418	0.716136	0.685983	0.521984	0.883121	1.000000	0.921391	0.500667	0.336783	...
concave points_mean	0.822529	0.293464	0.850977	0.823269	0.553695	0.831135	0.921391	1.000000	0.462497	0.166917	...
symmetry_mean	0.147741	0.071401	0.183027	0.151293	0.557775	0.602641	0.500667	0.462497	1.000000	0.479921	...
fractal_dimension_mean	-0.311631	-0.076437	-0.261477	-0.283110	0.584792	0.565369	0.336783	0.166917	0.479921	1.000000	...

Fig 5.2 Check for Multicollinearity



```
radius_mean                0
texture_mean               0
perimeter_mean             0
area_mean                  0
smoothness_mean            0
compactness_mean           0
concavity_mean             0
concave points_mean        0
symmetry_mean              0
fractal_dimension_mean     0
radius_se                  0
texture_se                 0
perimeter_se               0
area_se                    0
smoothness_se              0
compactness_se             0
concavity_se               0
concave points_se          0
symmetry_se                0
fractal_dimension_se       0
```

Fig 5.3 Checking of null values

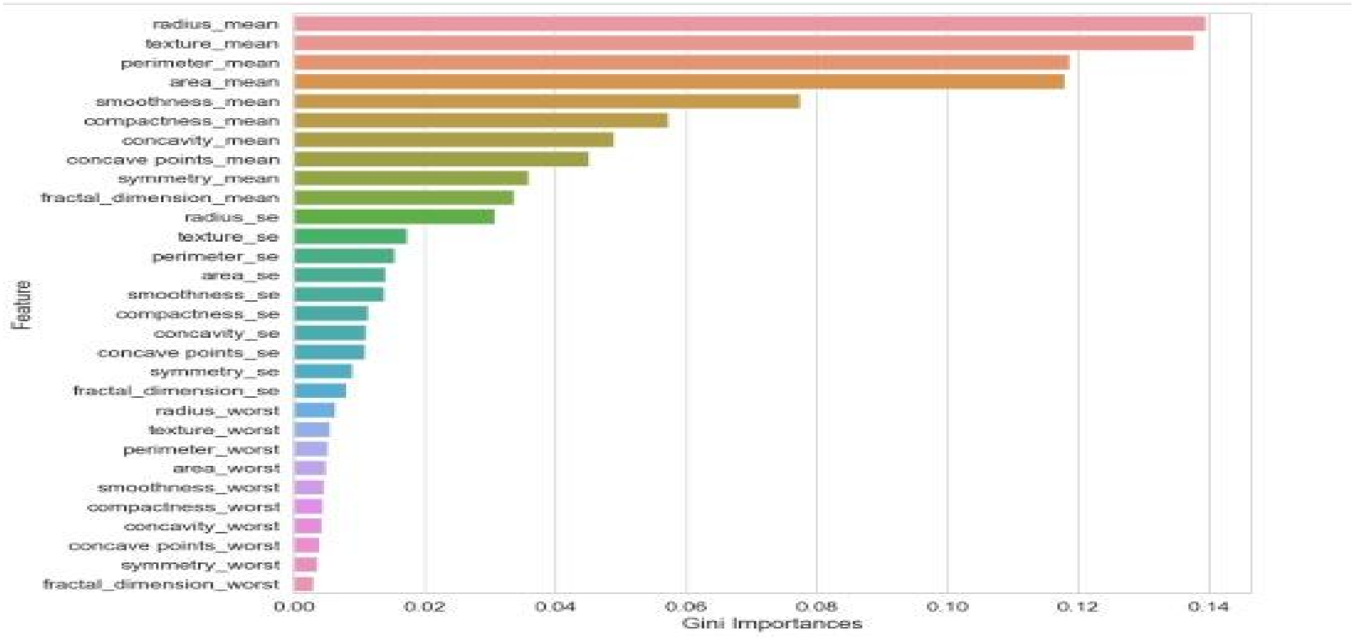
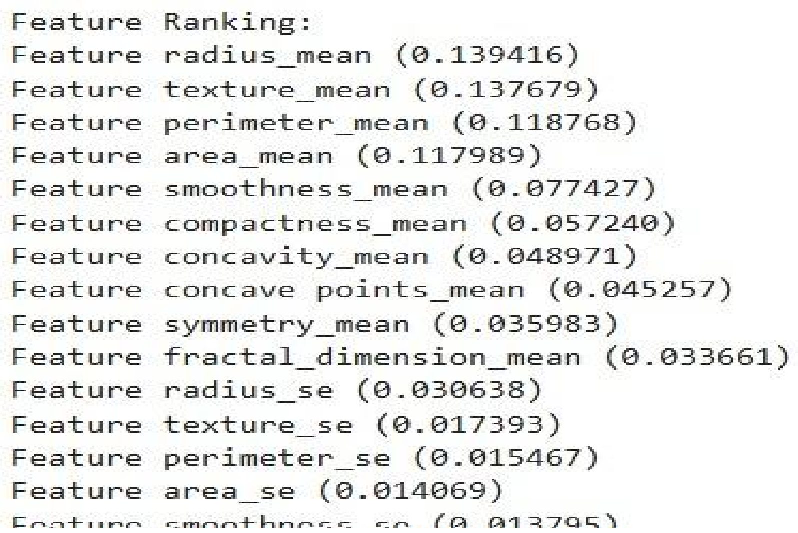


Fig 5.4 Feature selection chart



```
Feature Ranking:
Feature radius_mean (0.139416)
Feature texture_mean (0.137679)
Feature perimeter_mean (0.118768)
Feature area_mean (0.117989)
Feature smoothness_mean (0.077427)
Feature compactness_mean (0.057240)
Feature concavity_mean (0.048971)
Feature concave points_mean (0.045257)
Feature symmetry_mean (0.035983)
Feature fractal_dimension_mean (0.033661)
Feature radius_se (0.030638)
Feature texture_se (0.017393)
Feature perimeter_se (0.015467)
Feature area_se (0.014069)
Feature smoothness_se (0.013795)
```

Fig 5.5 Feature Ranking

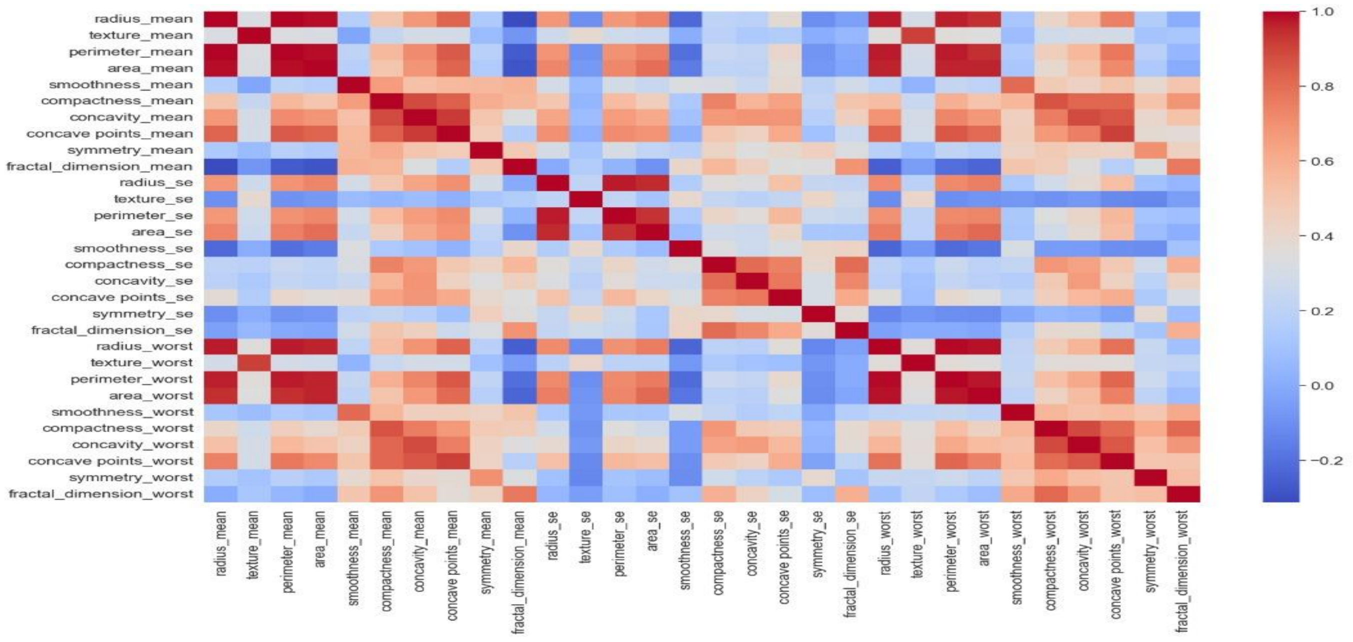


Fig 5.3 Heatmap plot

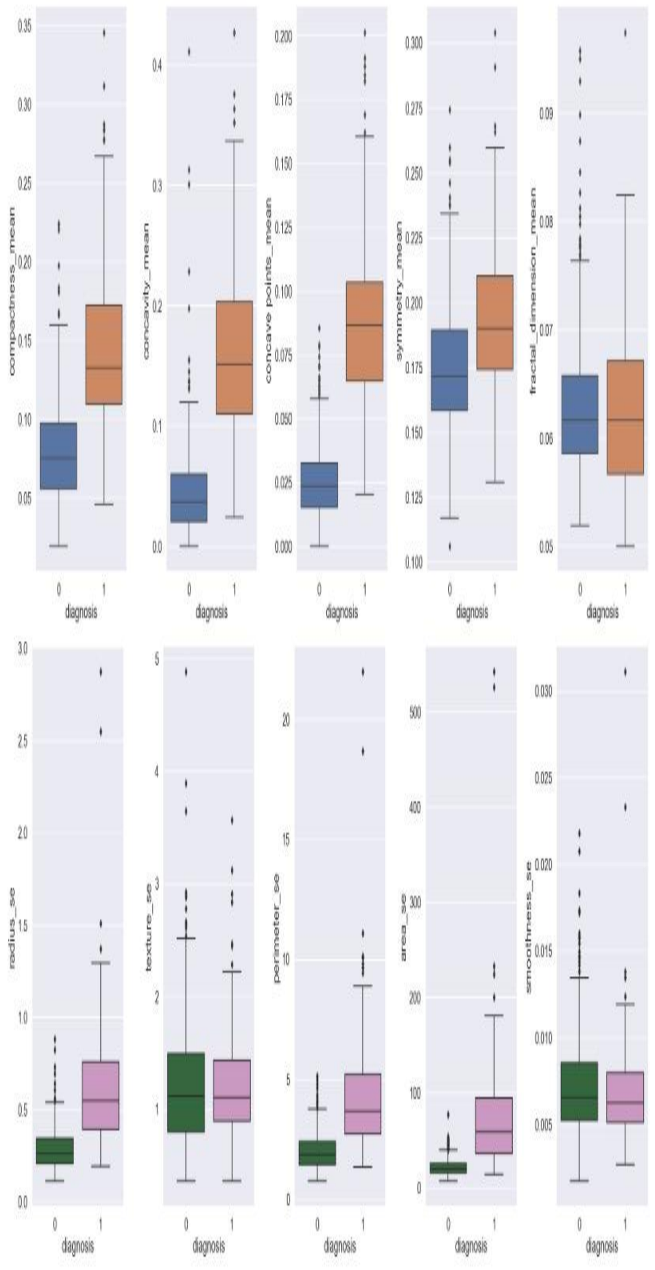


Fig 5.4Box plot

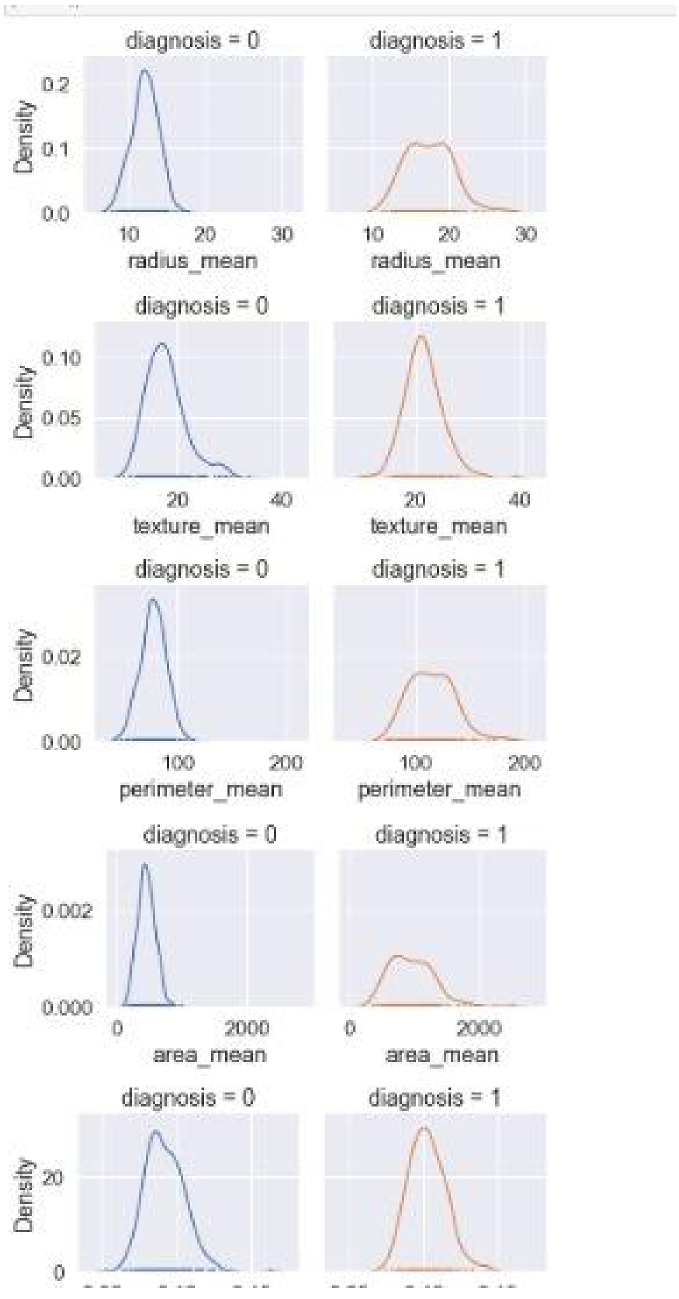
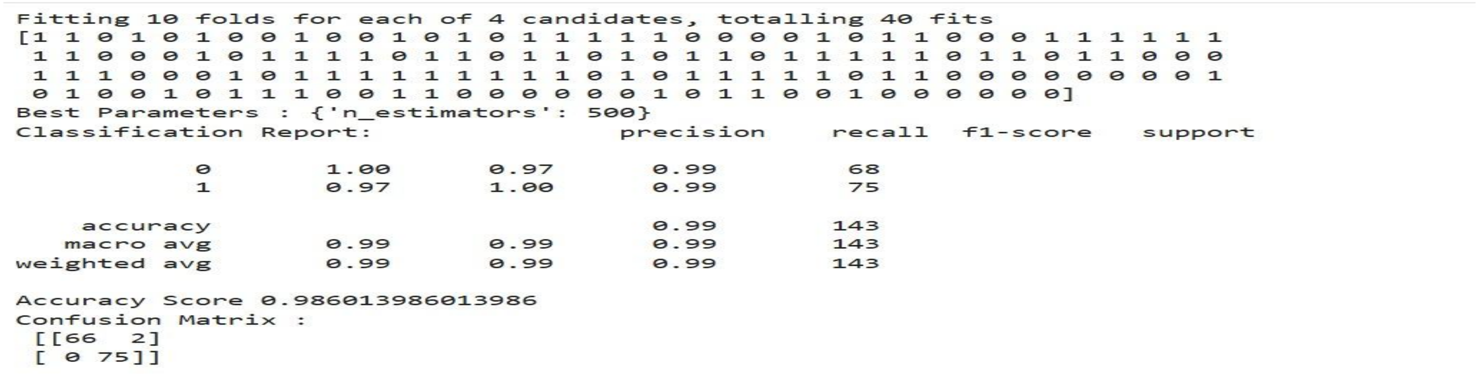


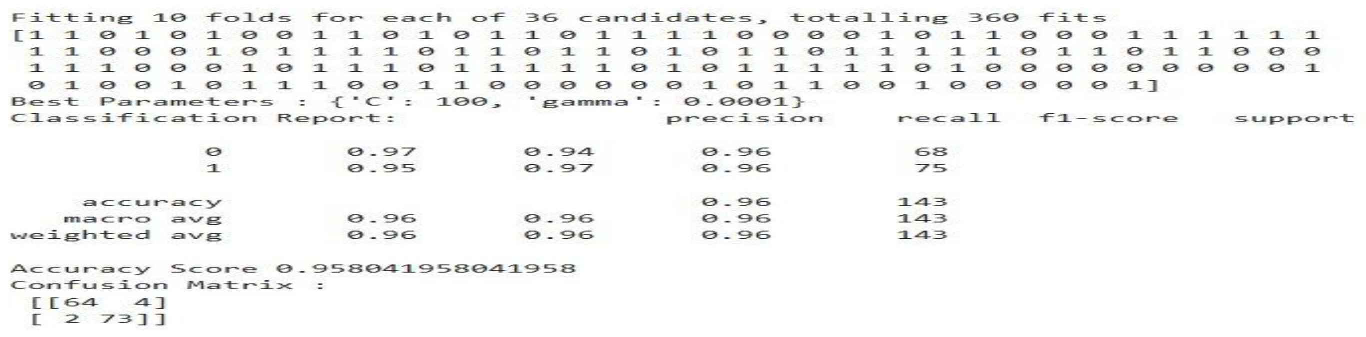
Fig 5.5 Distribution plot

Random Forest Model



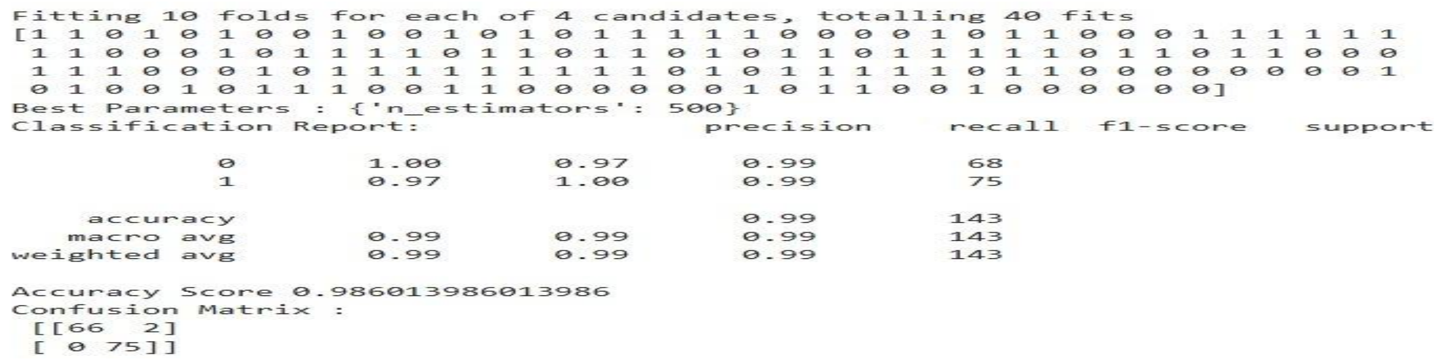
```
Fitting 10 folds for each of 4 candidates, totalling 40 fits
[1 1 0 1 0 1 0 0 1 0 0 1 0 1 0 1 1 1 1 1 0 0 0 0 1 0 1 1 0 0 0 1 1 1 1 1 1
 1 1 0 0 0 1 0 1 1 1 1 0 1 1 0 1 1 0 1 0 1 1 0 1 1 1 1 1 0 1 1 0 1 1 0 0 0
 1 1 1 0 0 0 1 0 1 1 1 1 1 1 1 1 1 0 1 0 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 1
 0 1 0 0 1 0 1 1 1 0 0 1 1 0 0 0 0 0 0 1 0 1 1 0 0 1 0 0 0 0 0 0]
Best Parameters : {'n_estimators': 500}
Classification Report:              precision    recall  f1-score   support

           0       1.00      0.97      0.99        68
           1       0.97      1.00      0.99        75

    accuracy                           0.99       143
   macro avg       0.99      0.99      0.99       143
weighted avg       0.99      0.99      0.99       143

Accuracy Score 0.986013986013986
Confusion Matrix :
 [[66  2]
 [ 0 75]]
```

SVC Model



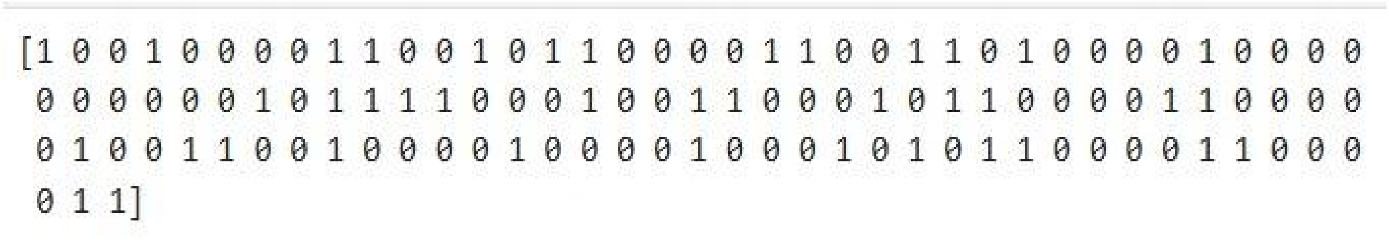
```
Fitting 10 folds for each of 36 candidates, totalling 360 fits
[1 1 0 1 0 1 0 0 1 1 0 1 0 1 1 0 1 1 1 1 0 0 0 0 1 0 1 1 0 0 0 1 1 1 1 1 1
 1 1 0 0 0 1 0 1 1 1 1 0 1 1 0 1 1 0 1 0 1 1 0 1 1 1 1 1 0 1 1 0 1 1 0 0 0
 1 1 1 0 0 0 1 0 1 1 1 0 1 1 1 1 1 0 1 0 1 1 1 1 1 0 1 0 0 0 0 0 0 0 0 0 1
 0 1 0 0 1 0 1 1 1 0 0 1 1 0 0 0 0 0 0 1 0 1 1 0 0 1 0 0 0 0 0 1]
Best Parameters : {'C': 100, 'gamma': 0.0001}
Classification Report:              precision    recall  f1-score   support

           0       0.97      0.94      0.96        68
           1       0.95      0.97      0.96        75

    accuracy                           0.96       143
   macro avg       0.96      0.96      0.96       143
weighted avg       0.96      0.96      0.96       143

Accuracy Score 0.958041958041958
Confusion Matrix :
 [[64  4]
 [ 2 73]]
```

XG Boost



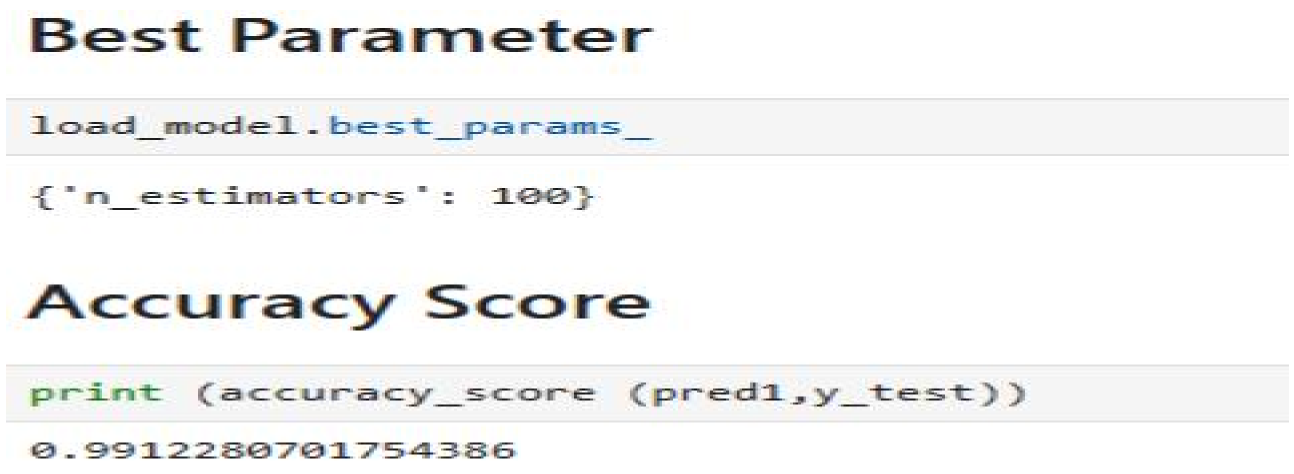
```
Fitting 10 folds for each of 4 candidates, totalling 40 fits
[1 1 0 1 0 1 0 0 1 0 0 1 0 1 0 1 1 1 1 1 0 0 0 0 1 0 1 1 0 0 0 1 1 1 1 1 1
 1 1 0 0 0 1 0 1 1 1 1 0 1 1 0 1 1 0 1 0 1 1 0 1 1 1 1 1 0 1 1 0 1 1 0 0 0
 1 1 1 0 0 0 1 0 1 1 1 1 1 1 1 1 1 0 1 0 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 1
 0 1 0 0 1 0 1 1 1 0 0 1 1 0 0 0 0 0 0 1 0 1 1 0 0 1 0 0 0 0 0 0]
Best Parameters : {'n_estimators': 500}
Classification Report:              precision    recall  f1-score   support

           0       1.00      0.97      0.99        68
           1       0.97      1.00      0.99        75

    accuracy                           0.99       143
   macro avg       0.99      0.99      0.99       143
weighted avg       0.99      0.99      0.99       143

Accuracy Score 0.986013986013986
Confusion Matrix :
 [[66  2]
 [ 0 75]]
```

Fig 5.8 Creating of Random Forest, XG Boost and SVC Models



```
[1 0 0 1 0 0 0 0 1 1 0 0 1 0 1 1 0 0 0 0 1 1 0 0 1 1 0 1 0 0 0 0 1 0 0 0 0
 0 0 0 0 0 0 1 0 1 1 1 1 0 0 0 1 0 0 1 1 0 0 0 1 0 1 1 0 0 0 1 1 0 0 0 0
 0 1 0 0 1 1 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 1 0 1 1 0 0 0 0 1 1 0 0 0
 0 1 1]
```

Fig 5.9 Predicted values



Best Parameter

```
load_model.best_params_
```

```
{'n_estimators': 100}
```

Accuracy Score

```
print (accuracy_score (pred1,y_test))
```

```
0.9912280701754386
```

Fig 5.10 best parameter and Accuracy score

CHAPTER 6

CONCLUSION

In conclusion, the cancer detection project successfully demonstrates the application of machine learning techniques to classify tumors as either Malignant or Benign based on various cell feature attributes. Through careful data preprocessing, including handling missing values, feature encoding, and normalization, the project ensures that the dataset is clean and ready for model training. Feature selection, driven by Random Forest's feature importance, plays a vital role in improving model accuracy by focusing on the most significant attributes.

The project utilizes multiple machine learning models, such as Support Vector Machine (SVM), Random Forest, and XGBoost, to predict cancer types. These models, each optimized through techniques like GridSearchCV, are evaluated based on their performance, with Random Forest and XGBoost proving to be highly effective in handling the dataset's complexities. By leveraging cross-validation and addressing class imbalance through methods like SMOTE, the project ensures reliable and generalizable results.

Overall, the project highlights the potential of machine learning in medical diagnostics, particularly in early cancer detection. The approach not only provides a framework for future research in cancer prediction but also opens avenues for further optimization and real-world application in healthcare.

REFERENCES

1. Breiman, L.(2001). "Random Forests." *Machine Learning*, 45(1), 5–32. This paper introduces the Random Forest algorithm, emphasizing its use in classification and regression tasks, which are critical for disease prediction models.
2. Chen, T., & Guestrin, C. (2016). "XGBoost: A Scalable Tree Boosting System." *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794.This study describes XGBoost, highlighting its speed and scalability, often applied in medical data analysis for its accuracy.
3. Deo, R. C. (2015). "Machine Learning in Medicine." *Circulation*, 132(20), 1920–1930. The article reviews various machine learning models, including SVM and Logistic Regression, for applications in medical diagnosis and disease prediction.
4. Sharma, A., & Virmani, J. (2022). "A Comprehensive Review on Machine Learning Algorithms for Disease Prediction." *Biomedical Signal Processing and Control*, 71, Article103144. This review compares ML algorithms, detailing their performance in healthcare applications, particularly for chronic disease prediction.
5. Sharma, A., & Virmani, J. (2022). "A Comprehensive Review on Machine Learning Algorithms for Disease Prediction." *Biomedical Signal Processing and Control*, 71, Article 103144.
6. Deo, R. C. (2015). "Machine Learning in Medicine." *Circulation*, 132(20), 1920–1930.
7. Lundberg, S. M., & Lee, S. I. (2017). "A Unified Approach to Interpreting Model Predictions."

Advances in Neural Information Processing Systems, 30, 4765–4774.

8. Haq, A. U., Li, J. P., Memon, M. H., Nazir, S., & Sun, N. (2018). "A Hybrid Intelligent System Framework for the Prediction of Heart Disease Using Machine Learning Algorithms." *Mobile Information Systems*, Article 3860146.
9. Zhang, Z., & Yang, L. (2021). "Machine Learning in Disease Prediction: Current Status and Future Directions." *Frontiers in Bioinformatics*, 1, Article 651159.

10. Goldstein, B. A., Navar, A. M., Pencina, M. J., & Ioannidis, J. P. A. (2017). "Opportunities and Challenges in Developing Risk Prediction Models with Machine Learning and Artificial Intelligence." *JAMA*, 320(7), 1797–1806.
11. World Health Organization. (2011). "Hemoglobin concentrations for the diagnosis of anemia and assessment of severity." *WHO Report No. WHO/NMH/NHD/MNM/11.1*.
12. Obaidy, M. A., et al. (2021). "Prevalence and Risk Factors of Anemia among Children Aged 5 months-12 years." *Mosul Journal of Nursing*, 9(1), 131-137.
13. Amin, N., & Habib, A. (2015). "Comparison of different classification techniques using WEKA for hematological data." *American Journal of Engineering Research*, 4(3), 55-61.
14. Jaiswal, M., Srivastava, A., & Siddiqui, T. J. (2019). "Machine learning algorithms for anemia disease prediction." *Recent Trends in Communication, Computing, and Electronics*. Springer Singapore.
15. Shilpa, S. A., Nagori, M., & Kshirsaga, V. (2011). "Classification of anemia using data mining techniques." *Swarm, Evolutionary, and Memetic Computing*, 113-121. Springer.
16. Sow, B., et al. (2020). "Assessing the relative importance of social determinants of health in malaria and anemia classification based on machine learning techniques." *Informatics for Health and Social Care*, 45(3), 229-241.