A

PROJECT REPORT

on

"BREAST CANCER PREDICTION"

Submitted in partial of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

SUBMITTED BY:
Chaitanya (A40219017)
Batch: 2019-2023

Shatakshi (A40320022)
Batch: 2020-2024

PROJECT GUIDE:
Mr. Pranay Sharma

Department of Computer Science & Engineering

Faculty of Engineering & Technology

P.D.M. University, Bahadurgarh.

pdm UNIVERSITY
Established under the Haryana Private Universities (Amendment) Act, 2015

CERTIFICATE OF COMPLETION

This is to certify that the Project work entitled "**Breast Cancer Prediction**" submitted by **Chaitanya and Shatakshi** in fulfilment for the requirements of the award of **Bachelor of Technology Degree in Computer Science & Engineering** at PDMU, Bahadurgarh, Haryana is an authentic work carried out by her under my supervision and guidance. To the best of my knowledge, the matter embodied in the project has not been submitted to any other University/ Institute for the award of any Degree.

Dr. Jasvinder Kaur
H.O.D, CSE Department
Faculty of Engineering & Technology

Post Box No. 15, Sector-3A, Sarai Aurangabad, Bahadurgarh-124507, Haryana, India. Phone No.: 01276-221700, 701. Fax: 01276-221714. info@pdm.ac.in, www.pdm.ac.in

DECLARATION BY THE CANDIDATES

We hereby declare that the work presented in this report entitled **"BREAST CANCER PREDICTION"**, in fulfilment of the requirement for the award of the degree Bachelor of Technology in Computer Science & Engineering, submitted in CSE Department, PDMU, Bahadurgarh, Haryana is an authentic record of our own work carried out during our degree under the guidance of **Dr. Jasvinder Kaur**.

The work reported in this has not been submitted by me for award of any other degree or diploma.

CHAITANYA
A40219017

SHATAKSHI
A40320022

ACKNOWLEDGEMENT

We express our sincere gratitude to **DR. JASVINDER KAUR** (H.O.D, Department of CSE) , for her valuable guidance and timely suggestions during the entire duration of our dissertation work, without which this work would not have been possible. We would also like to convey our deep regards to all other faculty members who have bestowed their great effort and guidance at appropriate times without which it would have been very difficult on our part to finish this work.

CHAITANYA
A40219017

SHATAKSHI
A40320022

ABSTRACT

Breast cancer is the most common malignancy among women, accounting for nearly 1 in 3 cancers diagnosed among women in the United States, and it is the second leading cause of cancer death among women. Breast Cancer occurs because of abnormal growth of cells in the breast tissue, commonly referred to as a Tumor. A tumor does not mean cancer - tumors can be benign (not cancerous), pre-malignant (pre-cancerous), or malignant (cancerous). Tests such as MRI, mammogram, ultrasound and biopsy are commonly used to diagnose breast cancer performed.

Women are seriously threatened by breast cancer with high morbidity and mortality. The lack of robust prognosis models results in difficulty for doctors to prepare a treatment plan that may prolong patient survival time. Hence, the requirement of time is to develop the technique which gives minimum error to increase accuracy. Four algorithm SVM, Logistic Regression, Random Forest and KNN which predict the breast cancer outcome have been compared in the paper using different datasets. All experiments are executed within a simulation environment and conducted in JUPYTER platform. Aim of research categorises in three domains. First domain is prediction of cancer before diagnosis, second domain is prediction of diagnosis and treatment and third domain focuses on outcome during treatment. The proposed work can be used to predict the outcome of different technique and suitable technique can be used depending upon requirement. This research is carried out to predict the accuracy. The future research can be carried out to predict the other different parameters and breast cancer research can be categorises on basis of other parameters.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

Women are seriously threatened by breast cancer with high morbidity and mortality. The lack of robust prognosis models results in difficulty for doctors to prepare a treatment plan that may prolong patient survival time. Hence, the requirement of time is to develop the technique which gives minimum error to increase accuracy. Four algorithm SVM, Logistic Regression, Random Forest and KNN which predict the breast cancer outcome have been compared in the paper using different datasets. All experiments are executed within a simulation environment and conducted in JUPYTER platform. Aim of research categorises in three domains. First domain is prediction of cancer before diagnosis, second domain is prediction of diagnosis and treatment and third domain focuses on outcome during treatment. The proposed work can be used to predict the outcome of different technique and suitable technique can be used depending upon requirement. This research is carried out to predict the accuracy. The future research can be carried out to predict the other different parameters and breast cancer research can be categorises on basis of other parameters.

The second major cause of women's death is breast cancer (after lung cancer). 246,660 of women's new cases of invasive breast cancer are expected to be diagnosed in the US during 2016 and 40,450 of women's death is estimated. Breast cancer is a type of cancer that starts in the breast. Cancer starts when cells begin to grow out of control. Breast cancer cells usually form a tumour that can often be seen on an x-ray or felt as a lump. Breast cancer can spread when the cancer cells get into the blood or lymph system and are carried to other parts of the body. The cause of Breast Cancer includes changes and mutations in DNA.

There are many different types of breast cancer and common ones include ductal carcinoma in situ (DCIS) and invasive carcinoma. Others, like phyllodes tumours and angiosarcoma are less common. There are many algorithms for classification of breast cancer outcomes. The side effects of Breast Cancer are – Fatigue, Headaches, Pain and numbness (peripheral neuropathy), Bone loss and osteoporosis. There are many algorithms for classification and prediction of breast cancer outcomes. The present paper gives a comparison between the performance of four classifiers: SVM , Logistic Regression , Random Forest and kNN which are among the most influential data mining algorithms.

It can be medically detected early during a screening examination through mammography or by portable cancer diagnostic tool. Cancerous breast tissues change with the progression of the disease, which can be directly linked to cancer staging. The stage of breast cancer (I–IV) describes how far a patient's cancer has proliferated. Statistical indicators such as tumour size, lymph node metastasis, and distant metastasis and so on are used to determine stages. To prevent cancer from spreading, patients have to undergo breast cancer surgery, chemotherapy, radiotherapy and endocrine. The goal of the research is to identify and classify Malignant and Benign patients and intending how to parametrize our classification techniques hence to achieve high accuracy. We are looking into many datasets and how further Machine Learning algorithms can be used to characterize Breast Cancer. We want to reduce the error rates with maximum accuracy. 10-fold cross validation test which is a Machine Learning Technique is used in JUPYTER to evaluate the data and analyse data in terms of effectiveness and efficiency.

CHAPTER 2: SOFTWARE REQUIREMENTS AND SPECIFICATIONS (SRS) DOCUMENTS

Objectives of the Project

Thus, the goal is to classify whether the breast cancer is benign or malignant and predict the recurrence and non-recurrence of malignant cases after a certain period. To achieve this, we have used machine learning classification methods to fit a function that can predict the discrete class of new input.

Identify data sources

The Breast Cancer datasets is available machine learning repository maintained by the University of California, Irvine. The dataset contains **569 samples of malignant and benign tumor cells**.

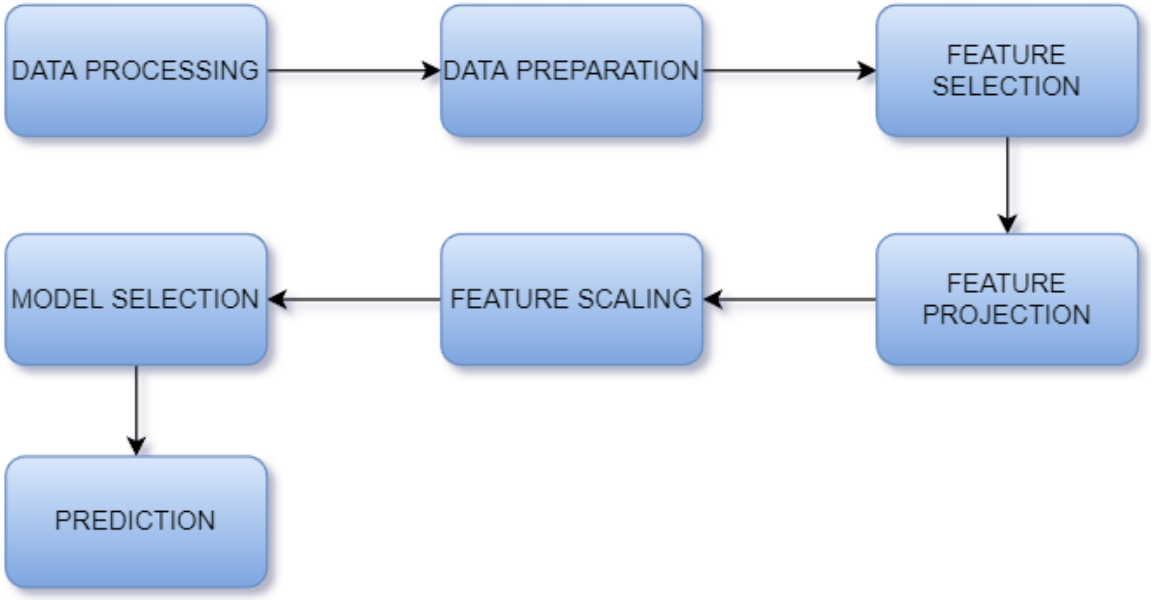
- The first two columns in the dataset store the unique ID numbers of the samples and the corresponding diagnosis (M=malignant, B=benign), respectively.
- The columns 3-32 contain 30 real-value features that have been computed from digitized images of the cell nuclei, which can be used to build a model to predict whether a tumor is benign or malignant.

Hardware and Software requirement

PLATFORM	Window 7 or above
SOFTWARE	Jupyter notebook,
PROCESSOR	Intel Core i3 and above
RAM	Minimum 8 GB

CHAPTER 3: METHODOLOGIES

3.1 Proposed Methodology



CHAPTER 4: PLANNING OF WORK

Phase 1 PRE-PROCESSING DATA

The first phase we do is to collect the data that we are interested in collecting for pre-processing and to apply classification and Regression methods. Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Real world data is often incomplete, inconsistent, and lacking certain to contain many errors. Data pre-processing is a proven method of resolving such issues. Data pre-processing prepares raw data for further processing. For pre-processing we have used standardization method to pre-process the UCI dataset. This step is very important because the quality and quantity of data that you gather will directly determine how good your predictive model can be. In this case we collect the Breast Cancer samples which are Benign and Malignant. This will be our training data.

Phase 2 DATA PREPRATION

Data Preparation, where we load our data into a suitable place and prepare it for use in our machine learning training. We'll first put all our data together, and then randomize the ordering.

Phase 3 FEATURES SELECTION

In machine learning and statistics, feature selection, also known as variable selection, attribute selection, is the process of selection a subset of relevant features for use in model construction. Data File and Feature Selection Breast Cancer Wisconsin (Diagnostic): - Data Set from Kaggle repository and out of 33 parameters we have selected about 10 parameters. Our target parameter is breast cancer diagnosis – malignant or benign. We have used Wrapper Method for Feature Selection. The important features found by the study are Radius mean, Texture mean, Perimeter mean, Area mean, concave points worst, Area worst, Area se, Texture worst, Texture mean, Smoothness worst, Smoothness mean, Radius mean, Symmetry mean.

Phase 4 FEATURE PROJECTION

Feature projection is transformation of high-dimensional space data to a lower dimensional space (with few attributes). Both linear and nonlinear reduction techniques can be used in accordance with the type of relationships among the features in the dataset.

Phase 5 FEATURE SCALING

Most of the times, your dataset will contain features highly varying in magnitudes, units, and range. But since, most of the machine learning algorithms use Euclidian distance between two data points in their computations. We need to bring all features to the same level of magnitudes. This can be achieved by scaling.

Phase 6 MODEL SELECTION

Supervised learning is the method in which the machine is trained on the data which the input and output are well labelled. The model can learn on the training data and can process the future data to predict outcome. They are grouped to Regression and Classification techniques. A regression problem is when the result is a real or continuous value, such as "salary" or "weight". A classification problem is when the result is a category like filtering emails spam" or "not spam". Unsupervised Learning: Unsupervised learning is giving away information to the

machine that is neither classified nor labelled and allowing the algorithm to analyse the given information without providing any directions. In unsupervised learning algorithm the machine is trained from the data which is not labelled or classified making the algorithm to work without proper instructions. In our dataset we have the outcome variable or Dependent variable i.e. Y having only two set of values, either M (Malign) or B (Benign). So, Classification algorithm of supervised learning is applied on it. We have chosen three different types of classification algorithms in Machine Learning. We can use a small linear model, which is a simple.

Phase 7 PREDICTION

Machine learning is using data to answer questions. So, Prediction, or inference, is the step where we get to answer some questions. This is the point of all this work, where the value of machine learning is real.

CHAPTER 4: SYSTEM DESIGN

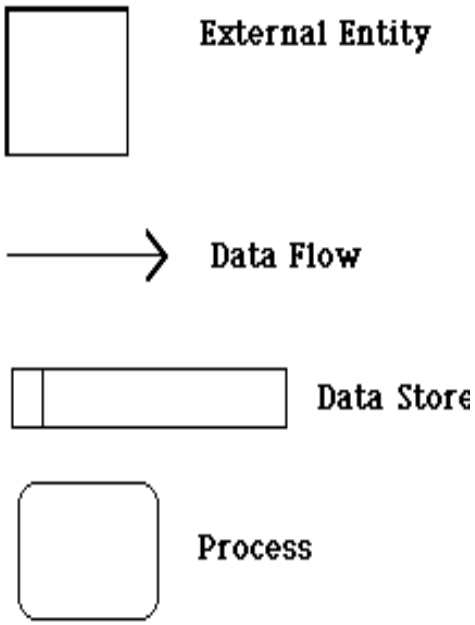
Data flow diagrams

A data flow diagram (DFD) illustrates how data is processed by a system in terms of inputs and outputs. As its name indicates its focus is on the flow of information, where data comes from, where it goes and how it gets stored.

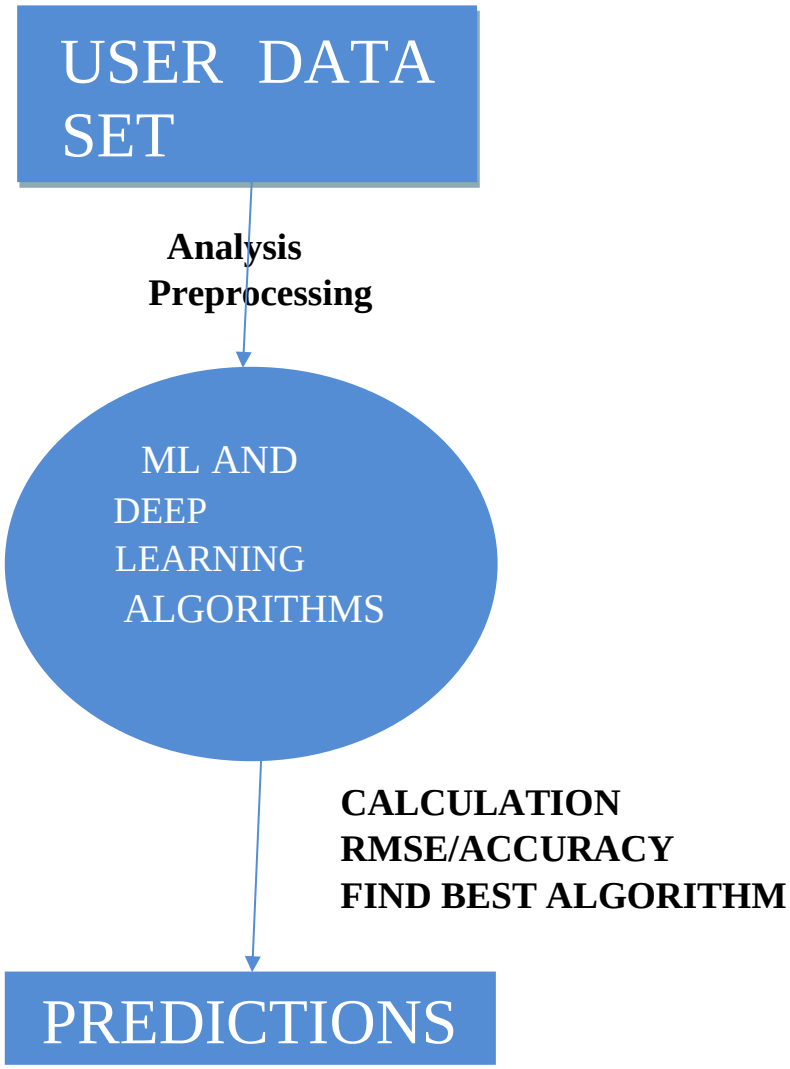
A context diagram is a top level (also known as "Level 0") data flow diagram. It only contains one process node ("Process 0") that generalizes the function of the entire system in relationship to external entities.

The context diagram is the highest level in a data flow diagram and contains only one process, representing the entire system. The process is given the numb external entities are shown on the context diagram, as well as major data flow to and from them. The diagram does not contain any data stores.

The following notations are used for any dfd:



A data flow diagram (DFD) illustrates how data is processed by a system in terms of inputs and outputs. The basic concepts related to smart systems and how machine learning techniques could add smart capabilities to many kinds of systems in almost any domain that you can imagine. Among other things, we learned that a typical workflow for a Machine Learning Project usually looks like the one shown in the image below:



ER DIAGRAMS

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs.

ER diagrams are related to data structure diagrams (DSDs), which focus on the relationships of elements within entities instead of relationships between entities themselves. ER diagrams also are often used in conjunction with data flow diagrams (DFDs), which map out the flow of information for processes or systems.

Uses of entity relationship diagrams

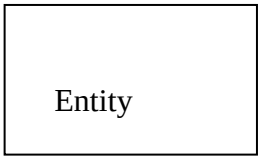
- Database design: ER diagrams are used to model and design relational databases, in terms of logic and business rules (in a logical data model) and in terms of the specific technology to be implemented (in a physical data model.) In software engineering, an ER diagram is often an initial step in determining requirements for an information systems project. It’s also later used to model a particular database or databases. A relational database has an equivalent relational table and can potentially be expressed that way as needed.
- Database troubleshooting: ER diagrams are used to analyze existing databases to find and resolve problems in logic or deployment. Drawing the diagram should reveal where it’s going wrong.
- Business information systems: The diagrams are used to design or analyze relational databases used in business processes. Any business process that uses fielded data involving entities, actions and interplay can potentially benefit from a relational database. It can streamline processes, uncover information more easily and improve results.
- Business process re-engineering (BPR): ER diagrams help in analyzing databases used in business process re-engineering and in modeling a new database setup.
- Education: Databases are today’s method of storing relational information for educational purposes and later retrieval, so ER Diagrams can be valuable in planning those data structures.
- Research: Since so much research focuses on structured data, ER diagrams can play a key role in setting up useful databases to analyze the data.

The components and features of an ER diagram

ER Diagrams are composed of entities, relationships and attributes. They also depict cardinality, which defines relationships in terms of numbers. Here’s a glossary:

Entity

A definable thing—such as a person, object, concept or event—that can have data stored about it. Think of entities as nouns. Examples: a customer, student, car or product. Typically shown as a rectangle.



`Entity type: A group of definable things, such as students or athletes, whereas the entity would be the specific student or athlete. Other examples: customers, cars or products.

Entity set: Same as an entitytype, but defined at a particular point in time, such as students enrolled in a class on the first day. Other examples: Customers who purchased last month, cars currently registered in Florida. A related term is instance, in which the specific person or car would be an instance of the entity set.

Entity categories: Entities are categorized as strong, weak or associative. A strong entity can be defined solely by its own attributes, while a weak entity cannot. An associative entity associates entities (or elements) within an entity set.

Entity keys: Refers to an attribute that uniquely defines an entity in an entity set. Entity keys can be super, candidate or primary. Super key: A set of attributes (one or more) that together define an entity in an entity set. Candidate key: A minimal super key, meaning it has the least possible number of attributes to still be a super key. An entity set may have more than one candidate key. Primary key: A candidate key chosen by the database designer to uniquely identify the entity set. Foreign key: Identifies the relationship between entities.

Relationship

How entities act upon each other or are associated with each other. Think of relationships as verbs. For example, the named student might register for a course. The two entities would be the student and the course, and the relationship depicted is the act of enrolling, connecting the two entities in that way. Relationships are typically shown as diamonds or labels directly on the connecting lines.

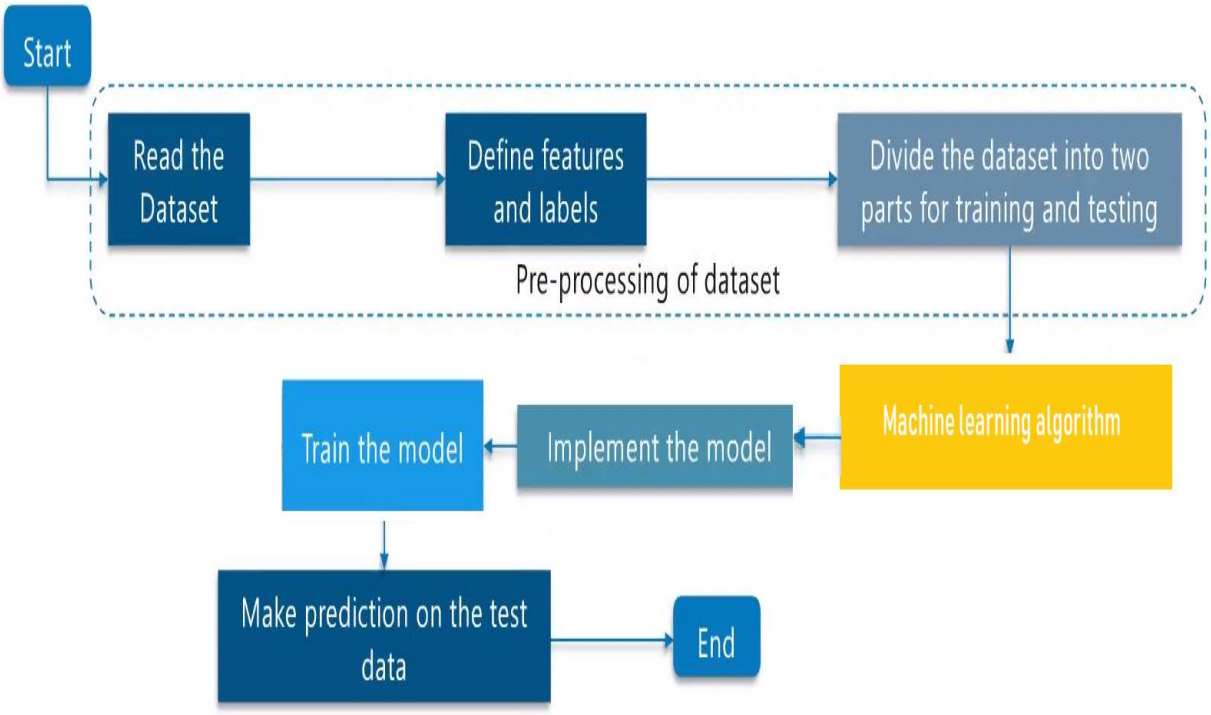


Fig: ER diagram of "Breast Cancer Prediction"

Flowchart

Flowchart Python is essentially the Python programming language in visual form. You write a program by setting up a flowchart. When you run the flowchart, the software compiles to python byte-code, so that you can easily import modules you write in Flowchart Python into standard Python programs. Flowchart Python does make a few additions to standard Python, which will be outlined on the home page eventually.

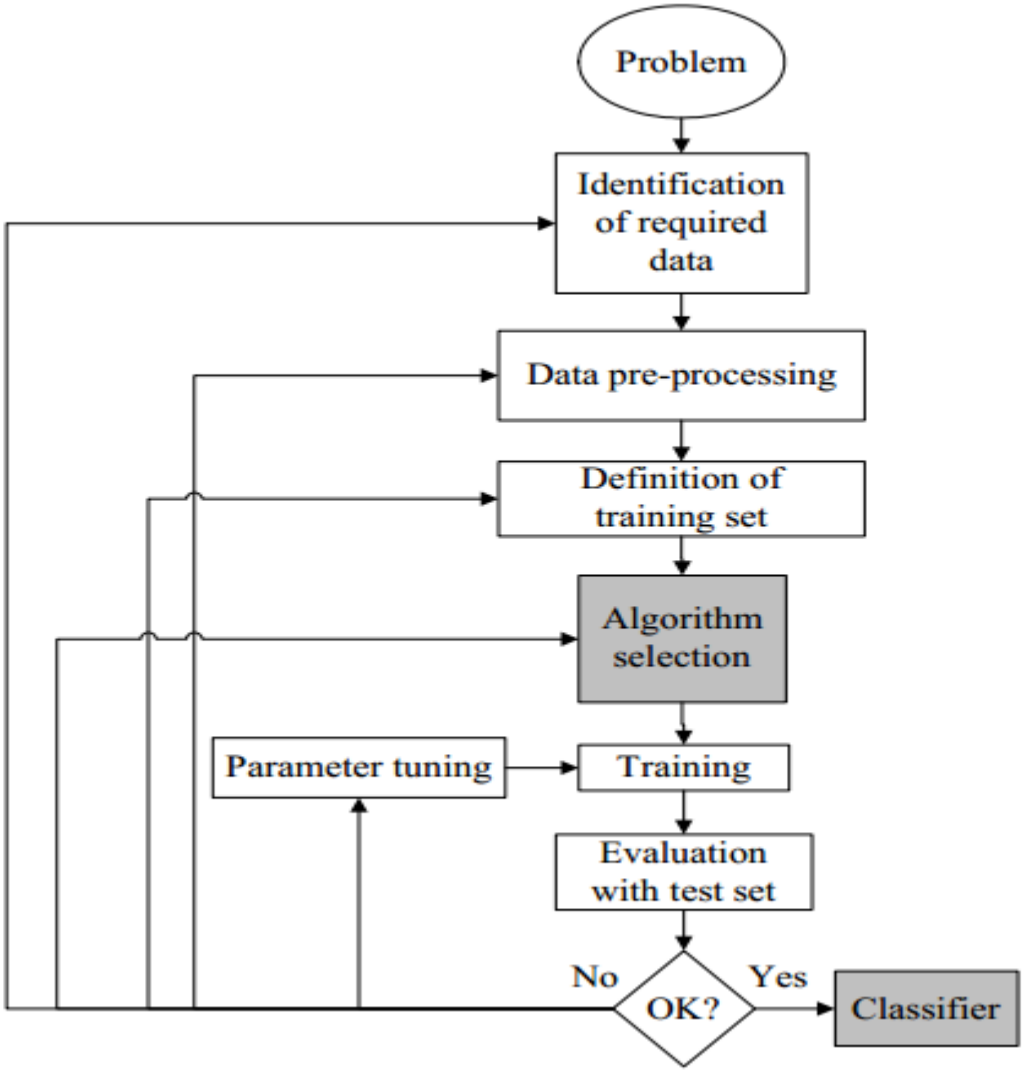


Fig: FLOW DIAGRAM OF MACHINE LEARNING

CHAPTER 5: CODING WITH RESULTS

 # loading libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

 # loading dataset and reading it

```
df = pd.read_csv('dataCleam.csv',index_col = False)
df.head( )
```

```
In [2]: df.head()
```

Out[2]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...	te
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...	

5 rows × 33 columns

#checking shape of dataset

```
df.shape
```

```
In [3]: df.shape
Out[3]: (569, 33)
```

#checking null values

```
df.isna().sum()
```

```
In [33]: df.isna().sum()

Out[33]: id                          0
         diagnosis                   0
         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
         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
         Unnamed: 32               569
         dtype: int64
```

#get a count ofthe number of Malignant (M) or Benign (B) cells

df.diagnosis.value_counts()

```
In [4]: #get a count ofthe number of Malignant (M) or Benign (B) cells
        df.diagnosis.value_counts()

Out[4]: B    357
        M    212
        Name: diagnosis, dtype: int64
```

#printing unique items of diagnosis column

df.diagnosis.unique()

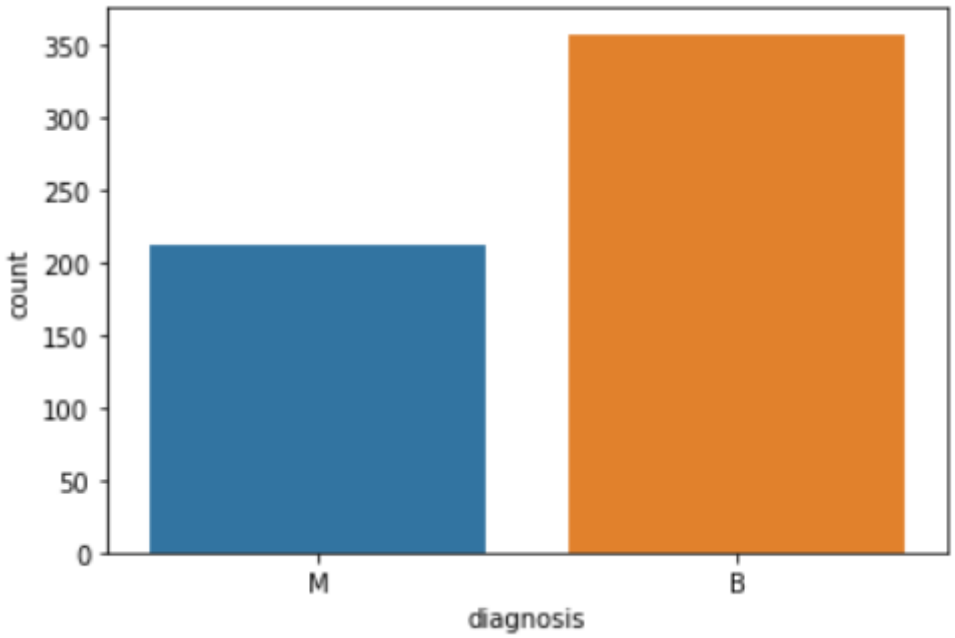
```
In [5]: #printing unique items of diagnosis coloumn
        df.diagnosis.unique()

Out[5]: array(['M', 'B'], dtype=object)
```

#visualize the count

sns.countplot(df.diagnosis,label="count")

plt.show()



#identifying datatypes

df.dtypes

```
id                          int64
diagnosis                  object
radius_mean               float64
texture_mean              float64
perimeter_mean            float64
area_mean                 float64
smoothness_mean           float64
compactness_mean          float64
concavity_mean            float64
concave points_mean       float64
symmetry_mean             float64
fractal_dimension_mean    float64
radius_se                 float64
texture_se                float64
perimeter_se              float64
area_se                   float64
smoothness_se             float64
compactness_se            float64
concavity_se              float64
concave points_se         float64
symmetry_se               float64
fractal_dimension_se      float64
radius_worst              float64
texture_worst             float64
perimeter_worst           float64
area_worst                float64
smoothness_worst          float64
compactness_worst         float64
concavity_worst           float64
concave points_worst      float64
symmetry_worst            float64
fractal_dimension_worst   float64
Unnamed: 32               float64
dtype: object
```

#encode the categorical data values

from sklearn.preprocessing import LabelEncoder

labelEncoder_Y=LabelEncoder()

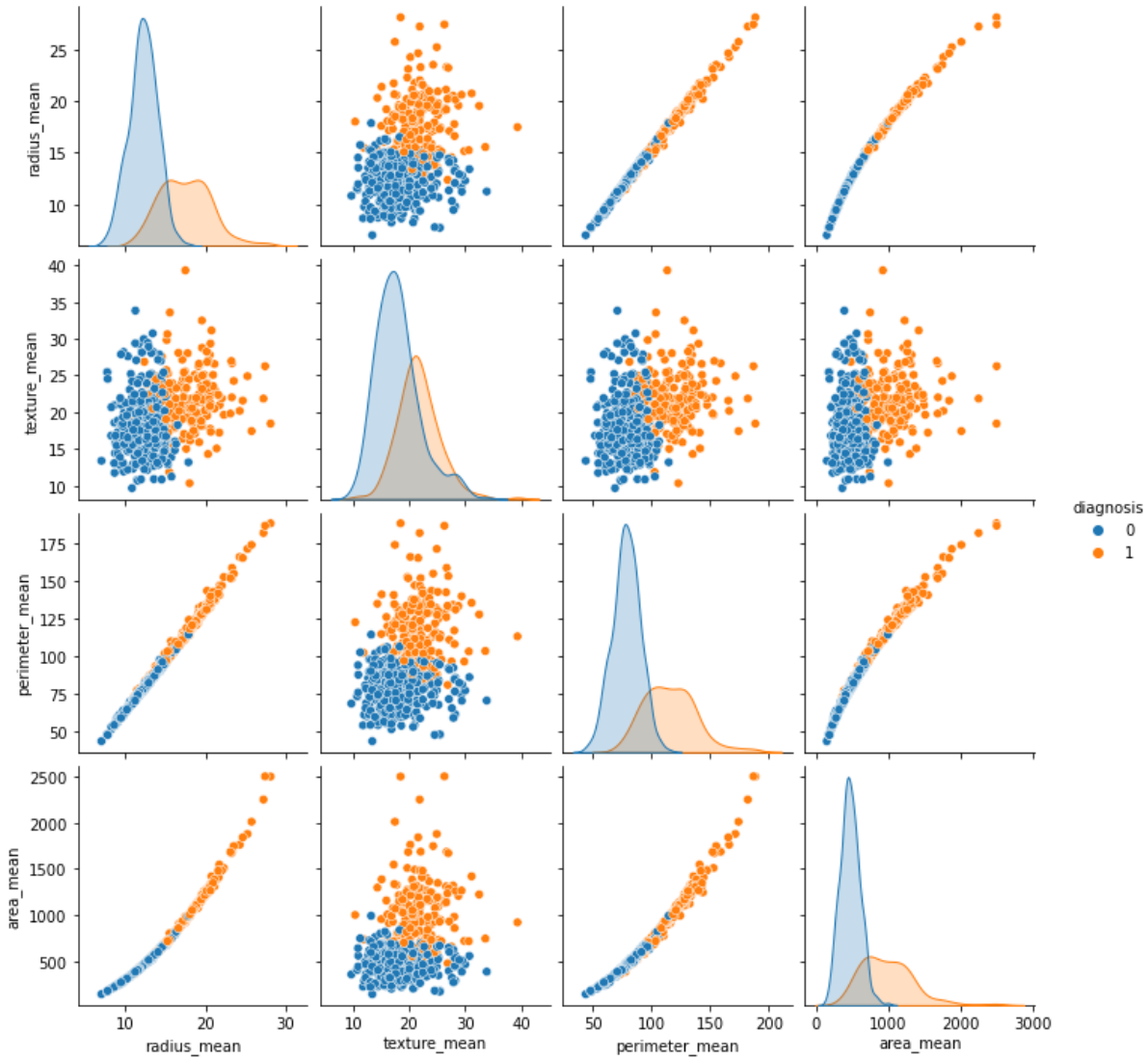
df.iloc[:,1]=labelEncoder_Y.fit_transform(df.iloc[:,1].values)

 # relationship between different features with diagnosis column

#create a pair plot

sns.pairplot(df.iloc[:,1:6],hue="diagnosis")

plt.show()



#get the correlation of the columns

df.iloc[:,1:12].corr()

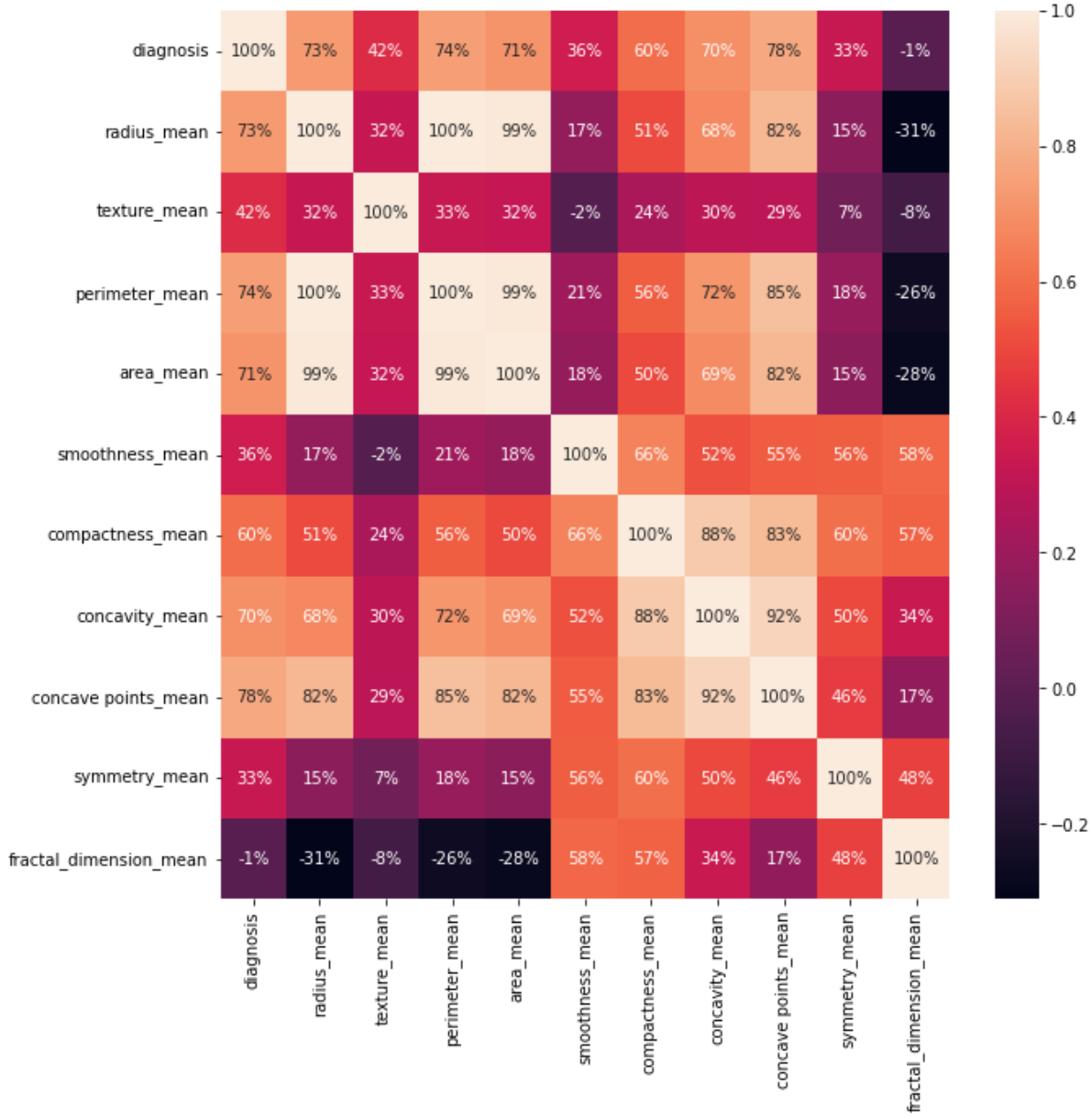
```
In [40]: #get the correlation of the columns
         df.iloc[:,1:12].corr()
```

Out[40]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	conc points_m
diagnosis	1.000000	0.730029	0.415185	0.742636	0.708984	0.358560	0.596534	0.696360	0.776
radius_mean	0.730029	1.000000	0.323782	0.997855	0.987357	0.170581	0.506124	0.676764	0.822
texture_mean	0.415185	0.323782	1.000000	0.329533	0.321086	-0.023389	0.236702	0.302418	0.293
perimeter_mean	0.742636	0.997855	0.329533	1.000000	0.986507	0.207278	0.556936	0.716136	0.850
area_mean	0.708984	0.987357	0.321086	0.986507	1.000000	0.177028	0.498502	0.685983	0.823
smoothness_mean	0.358560	0.170581	-0.023389	0.207278	0.177028	1.000000	0.659123	0.521984	0.553
compactness_mean	0.596534	0.506124	0.236702	0.556936	0.498502	0.659123	1.000000	0.883121	0.831
concavity_mean	0.696360	0.676764	0.302418	0.716136	0.685983	0.521984	0.883121	1.000000	0.921
concave points_mean	0.776614	0.822529	0.293464	0.850977	0.823269	0.553695	0.831135	0.921391	1.000
symmetry_mean	0.330499	0.147741	0.071401	0.183027	0.151293	0.557775	0.602641	0.500667	0.462
fractal_dimension_mean	-0.012838	-0.311631	-0.076437	-0.261477	-0.283110	0.584792	0.565369	0.336783	0.166

#visualize the correlation

```
plt.figure(figsize=(10,10))
sns.heatmap(df.iloc[:,1:12].corr(), annot=True,fmt=".0%")
plt.show()
```



#Split the data set into independent(x) and dependent (y) data sets

```
x=df.iloc[:,2:31].values
y=df.iloc[:,1].values
```

#split the data set into 75% training and 25% testing

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=0.25,random_state=0)
```

```
#scale the data(feature scaling)
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
x_train=sc.fit_transform(x_train)
x_test=sc.fit_transform(x_test)

#create a function for the models
def models(x_train,y_train):
  #Logistic Regression Model
  from sklearn.linear_model import LogisticRegression
  log=LogisticRegression(random_state=0)
  log.fit(x_train,y_train)

  #Decision Tree
  from sklearn.tree import DecisionTreeClassifier
  tree=DecisionTreeClassifier(criterion='entropy',random_state=0)
  tree.fit(x_train,y_train)

  #Random Forest Classifier
  from sklearn.ensemble import RandomForestClassifier
  forest = RandomForestClassifier(n_estimators=10,criterion="entropy",random_state=0)
  forest.fit(x_train,y_train)

  #Print the models accuracy on the training data
  print("[0]Logistic Regression Training Accuracy:",log.score(x_train,y_train))
  print("[1]Decision Tree Classifier Training Accuracy:",tree.score(x_train,y_train))
  print("[2]Random Forest Classifier Training Accuracy:",forest.score(x_train,y_train))
  return log,tree,forest
```

#Getting all of the models

model = models(x_train,y_train)

```
In [17]: #Getting all of the models
         model = models(x_train,y_train)

         [0]Logistic Regression Training Accuracy: 0.9906103286384976
         [1]Decision Tree Classifier Training Accuracy: 1.0
         [2]Random Forest Classifier Training Accuracy: 0.9953051643192489
```

#test model accuracy on confusion matrix

from sklearn.metrics import confusion_matrix

for i in range(len(model)):

print("Model ", i)

cm =confusion_matrix(y_test,model[i].predict(x_test))

TP=cm[0][0]

TN=cm[1][1]

FN=cm[1][0]

FP=cm[0][1]

print(cm)

print("Testing Accuracy = ", (TP+TN) / (TP+TN+FN+FP))

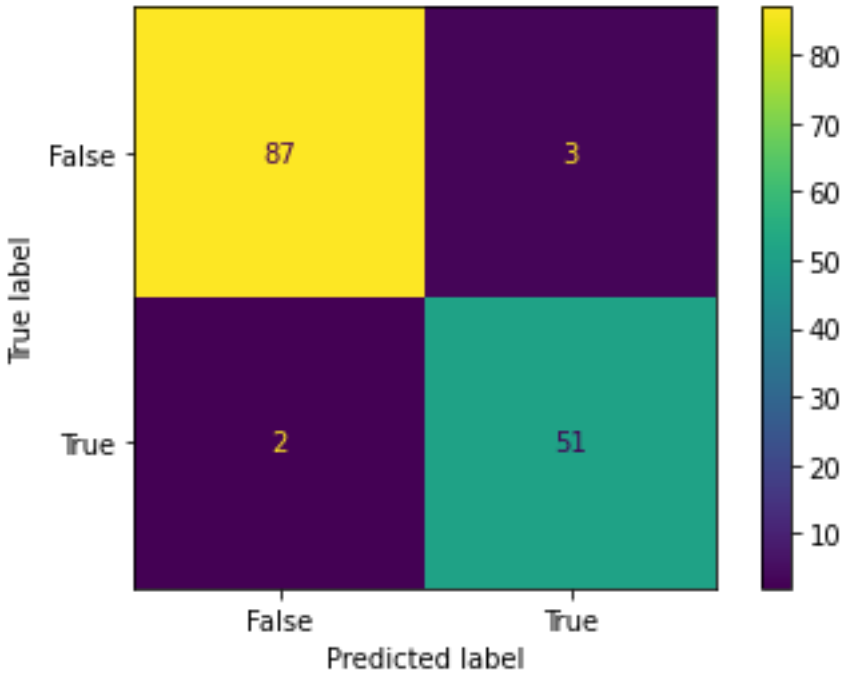
print()

```
Model  0
[[86  4]
 [ 3 50]]
Testing Accuracy =  0.951048951048951

Model  1
[[83  7]
 [ 2 51]]
Testing Accuracy =  0.9370629370629371

Model  2
[[87  3]
 [ 2 51]]
Testing Accuracy =  0.965034965034965
```

```
#printing the confusion matrix
from sklearn import metrics
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = cm, display_labels = [False, True])
cm_display.plot()
plt.show()
```



#show another way to get metrics of the models

```
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
for i in range(len(model) ):
  print("Model ",i)
  print( classification_report(y_test,model[i].predict(x_test)))
  print( accuracy_score(y_test,model[i].predict(x_test)))
  print()
```

```
Model  0
              precision    recall  f1-score   support

           0       0.97      0.96      0.96        90
           1       0.93      0.94      0.93        53

    accuracy                           0.95       143
   macro avg       0.95      0.95      0.95       143
weighted avg       0.95      0.95      0.95       143

0.951048951048951

Model  1
              precision    recall  f1-score   support

           0       0.98      0.92      0.95        90
           1       0.88      0.96      0.92        53

    accuracy                           0.94       143
   macro avg       0.93      0.94      0.93       143
weighted avg       0.94      0.94      0.94       143

0.9370629370629371

Model  2
              precision    recall  f1-score   support

           0       0.98      0.97      0.97        90
           1       0.94      0.96      0.95        53

    accuracy                           0.97       143
   macro avg       0.96      0.96      0.96       143
weighted avg       0.97      0.97      0.97       143

0.965034965034965
```

#print the prediction of random forest classifier model

pred=model[1].predict(x_test)

print(pred)

print()

print(y_test)

```
In [22]: #print the prediction of random forest classifier model
         pred=model[1].predict(x_test)
         print(pred)
         print()
         print(y_test)
```

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

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

BIBLOGRAPHY

- Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems: Aurélien Géron
- https://www.kaggle.com/