Summer Training Report

Entitled

Detection of Breast Cancer with Python

By

Neha Singh

HM- 24, Roll number : 0943

MBA (Hospital and Health Management)

2019-21

AKNOWLEDGEMENT

I am thankful to IIHMR University, Jaipur for acknowledging my potential report on ‘Detection of Breast Cancer with Python’.

I acknowledge my gratitude to my mentor Asst. Prof. SP Chattopadhyay who provided me every possible support. His unflinching help and encouragement was a constant source of inspiration in completion of this report by applying my knowledge.

I avail this opportunity to also convey my sincere gratitude to the co-operation and help received from my parents, brother and friends who helped and assisted me in carrying and completing this report.

At the time of preparing this report, I referred various websites and reviewed literature that helped me to get acquainted with new topics.

S.no.	Table of Contents	Page no.

1.	**Introduction**	**1**
2.	**Research Objective**	**2**
3.	**Methodology**	**2**
	Dataset	**2**
	Data Exploration and Cleaning	**3**
	Data Visualization	**7**
	Model Selection	**12**
	Result and Discussion	**16**
	Conclusion	**17**
4.	**References**	**18**
5.	**Annexure**	**19**

ACRONYMS

1. **WBCD –** Wisconsin Breast Cancer Diagnostics
2. **LR –** Logistic Regression
3. **KNN –** K Nearest Neighbour
4. **SVC –** Support Vector Classifier
5. **SVM –** Support Vector Machine
6. **GaussianNB –** Gaussian Naïve Bayes
7. **DT –** Decision Tree
8. **DCE-MRI –** Dynamic contrast-enhanced magnetic resonance imaging
9. **ANN –** Artificial Neural Networks
10. **M –** Malignant
11. **B –** Benign

Detection of Breast Cancer with Python

1. Introduction

Cancer is named after its originating body part; thus breast cancer refers to sudden growth of breast tissue cells. This forms a lump or mass of extra tissue, also known as tumor. Tumors are of two types viz. cancerous (malignant) or non-cancerous (benign). The term breast cancer refers to malignant tumor. Women between 40 and 55 years of age face the greatest risk of death from breast cancer, and are ranked second greatest among women **[1]**. With a greater focus on diagnostic strategies and successful diagnosis, the mortality rate has substantially decreased **[2]**. The various signs and symptoms that may occur in breast cancer are: a lump or thickening relative to the surrounding breast tissue, changes in breast size , shape or appearance, skin changes such as dimpling, appearance of inverted nipple, redness of the skin over the breast or peeling, scaling, crusting of the pigmented region around areola or breast skin **[3]**. Taking symptoms into consideration, medical diagnosis is gradually increasing the use of classifier system. Undoubtedly, evaluating patients dataset by expert decision on it is an important factor but however, systems and artificial intelligence techniques improves diagnosis on a higher level. It will not only minimize the possible error, but also examines the medical report precisely in short period of time.

Machine learning methods have been commonly used in health-care systems for decades. With the advent of emerging technology, big data can be accessed and processed faster, for example electronic medical records **[4]**. Big data can be accessed and analyzed quicker with the emergence of digital technologies, for example, electronic medical records. The healthcare system driven by technologies is an important asset. It assists professionals to diagnose patients accurately and provide more meaningful benchmark. Machine learning now handles some complex manual work in health industry like text and voice industry, which identify patients' emotion corresponding to health professional responses **[5-6]**.

In recent decades, many data mining and machine learning techniques have been developed and used to diagnose and identify breast cancer **[7-9]**. Numerous data mining and machine learning techniques have been developed and used in recent decades to diagnose and classify breast cancer. Mammography film pre-processing helps improve the visibility and strength distribution of peripheral regions. There are several methods to assist preprocessing. Feature extraction is next stage in detection of breast cancer as it helps in differentiating benign tumor from malignant. After this, image properties of breast viz. smoothness, depth, regularity and coarseness are extracted by segmentation **[10]**. The classification stage is a complex problem of optimization and several machine learning techniques have been used by researchers while solving classification problem as promising is the veracity of the machine learning technology.

Nowadays, the dependency on machine learning is growing until it will be adapted in services. But, machine learning is yet a field unexplored in many ways with barriers and often required expert knowledge. In the following sections, literature review and a comprehensive explanation to methodology applied to breast cancer detection using python will be given. While in search of best and accurate algorithm classification result, variable quality result of data affects the classification result. So, to test the machine learning technique in dataset, a benchmark dataset of Wisconsin breast cancer diagnosis (WBCD) is used in application **[11-12]**. There are also other datasets of the benchmark for breast cancer **[13]**, for example used in this paper Wisconsin Breast Cancer (Diagnostics) (WBCD) **[14]**.

2. Research Objective

With the help of python, the aim is to classify benign and malignant breast cancer tumors. The emphasis is on using the Logistic Regression (LR), K-Neighbours Classifier, Support Vector Classifier linear (SVC linear), Gaussian Naive Bayes (GaussianNB), and Decision Tree Classifier (DT) for evaluating and selecting the most effective detection model.

3. Methodology

The methodology is aimed at analyzing the most useful feature of malignant and benign tumor prediction. This can help visualize the general trend in selecting suitable model. With the aid of python, the goal is to distinguish benign and malignant breast cancer tumors. The focus is on using Logistic Regression, K-Neighbours Classifier, Support Vector Classifier linear (SVC linear), Gaussian Naive Bayes (GaussianNB), and Decision Tree Classifier (DT).

1. Dataset

The Wisconsin Breast Cancer Diagnostics (WBCD) dataset is used in this paper and is obtained from the UCI Machine Learning Repository[14]. It was created by Dr. William H. Wolberg, a physician at the Madison, Wisconsin, USA, University of Wisconsin Hospital. The data consists of 569 patients and 32 characteristics. These characteristics formed 32 columns in the dataset. Ten highlights of these characteristics are as per following:

- Radius
- Texture
- Perimeter
- Area
- Smoothness
- Compactness

- Concavity
- Concave points
- Symmetry
- Fractal dimension- the mean, standard error, and worst (mean of the three largest values) of all the features or characteristics.

2. Data Exploration and Cleaning

In this paper the software used to work on dataset of interest is Jupyter Notebook. To begin with, data exploration and cleaning is done via following steps in Jupyter Notebook:

2.1. Import librabries

Necessary libraries viz**. numpy, pandas**, **matplotlib.pyplot** and **seaborn** are imported.

```
In [1]: #import libraries
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
```

Figure 1 Import libraries

2.2. Load data

Download the breast cancer data.csv file from https://www.kaggle.com/uciml/breast-cancer-wisconsin-data and read this file using the read csv command pandas. Print the first 5 rows of DataFrame resulting. Each row of output data reflects a patient who has cancer or does not.

```
In [2]: #Load the data
        df = pd.read_csv('data.csv')
        df.head()
```

Out[2]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280

5 rows × 33 columns

Figure 2 First five rows of loaded data

2.3. List the rows and columns

Discover and study the data form by counting the number of rows and columns.

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

Figure 3 Number of rows and columns

There are 569 rows and 33 columns which represent 569 patients with 33 data points or features for individual patient in this dataset.

2.4. Count of columns containing empty values

Data sets are not perfect and DataFrame from dataset might contain some missing values in any column or row which can mislead the interpretion.

For every missing value, **pandas** add NaN at its place. So, fetch the count of all the column and row from the dataset that contain empty values viz. **NaN, NAN** or **na**. NaN occurs in case of any missing value.

```
In [4]: #Count the empty (NaN, NAN, na) values in each column
        df.isna().sum()

Out[4]: 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
```

Figure 4 Count of empty values in each

From the output it can be seen that only the column named ‘Unnamed: 32’, contains 569 empty values. Thus, the column ‘Unnamed: 32’ is of no use.

2.5. Dropping column ‘Unnamed: 32’ and creating new count of dataset

Since the column ‘Unnamed: 32’ is empty and contains no value, it is removed or dropped from the original dataset by using pandas **dropna** method function.

```
In [5]: df = df.dropna(axis =1 )
```

Figure 5 Function to drop column of empty value

Here **axis = 1** means dropping column containing any missing value.

Number of DataFrame df rows and columns will be counted using pandas **shape**

```
In [6]: df.shape

Out[6]: (569, 32)
```

Figure 6 Amount of existing DataFrame rows and columns;

DataFrame's current count comprises **569 rows**, and **32 columns**.

2.6. Count number of malignant and benign tumors

Pandas **value_counts** feature is used to count the number of patients diagnosed with either malignant (M) or benign (B) tumour.

```
In [7]: #Get a count of the number of 'M' & 'B' cells
        df['diagnosis'].value_counts()

Out[7]: B    357
        M    212
        Name: diagnosis, dtype: int64
```

Figure 7 Count of ‘M’ and ‘B’

So, the number of patients diagnosed with benign tumor is **357** and those with malignant tumor are **212**.

2.7. Listing columns and their data type

This is done to check the type of data present in the dataset. In case of any categorical data, the data is changed in dummy numeric. This is done because categorical data will mislead the interpretation of data.

The data type of columns is listed by using pandas **dtypes**.

```
In [9]: #Look at the data types
        df.dtypes

Out[9]: 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
        dtype: object
```

Figure 8 Data type of columns

In data types it can be seen that all the features/columns are numerical except for 'diagnosis' which is a categorical data and is defined in python as 'object.'

2.8. Encoding categorical data

To make sure that the learning algorithm interprets the malignant and benign tumor correctly, the categorical string values is converted into integers.

The categorical data present in column ‘diagnosis’ is encoded/ transformed from M and B to **1** and **0** by using **LabelEncoder** from **sklearn.preprocessing.**

```
In [10]: #Encoding categorical data values (
         from sklearn.preprocessing import LabelEncoder
         labelencoder_Y = LabelEncoder()
         df.iloc[:,1]= labelencoder_Y.fit_transform(df.iloc[:,1].values)
         print(labelencoder_Y.fit_transform(df.iloc[:,1].values))#Encoding categorical data
```

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

Figure 9 Encoding categorical data to numerical data

3. Visualisation of Data

Data visualization is the technique for interpreting data by putting it in a graphic form to communicate information interactively and effectively, such that patterns, trends and associations of data that would otherwise not be observed can be presented in broad data sets. It removes the noise from the data and highlights the useful information. As visualization makes it easier to detect patterns, trends and outliers, and provides clear, better and reliable result, it is implemented in this paper by creating count plot, pair plot and heat map.

In this work, data visualization is done with the help of **seaborn** library.

3.1. Count Plot

The plot will represent the class distribution of diagnosed malignant and benign patients. Here we have 212 malignant diagnosed patients i.e. around 38% of the data and, 357 i.e. 62% of patients diagnosed with benign tumor.

Count plot visualization of above data in python is done by using **seaborn countplot** function.

```
In [8]: #Visualize this count
        sns.countplot(df['diagnosis'],label="Count")

Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0x233b7e1f848>
```

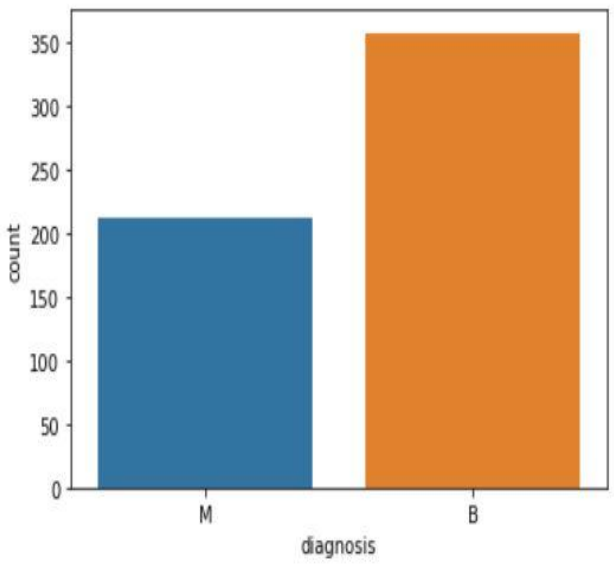

Figure 10 Count Plot for Malignant & Benign patients

3.2. Pair Plot

As the dataset contain many variables, and relationship between each and every variable is to be analysed, a pair plot is used to visualise the data further. It shows the data as a collection of points. One variable 's location in the same data row matches the value of another variable. Each value is a position on either the vertical or horizontal dimension indicates its correlation. It allows both, distribution of single variables and relationships between two variables. It is an effective method to identify trends for analysis.

To implement pair plots in python **seaborn** is used and is made by using seaborn **pairplot** function.

```
In [11]: #create a pair plot
         sns.pairplot(df, hue="diagnosis")

Out[11]: <seaborn.axisgrid.PairGrid at 0x233bacb9fc8>
```

Figure 11.1 Function to create pairplot

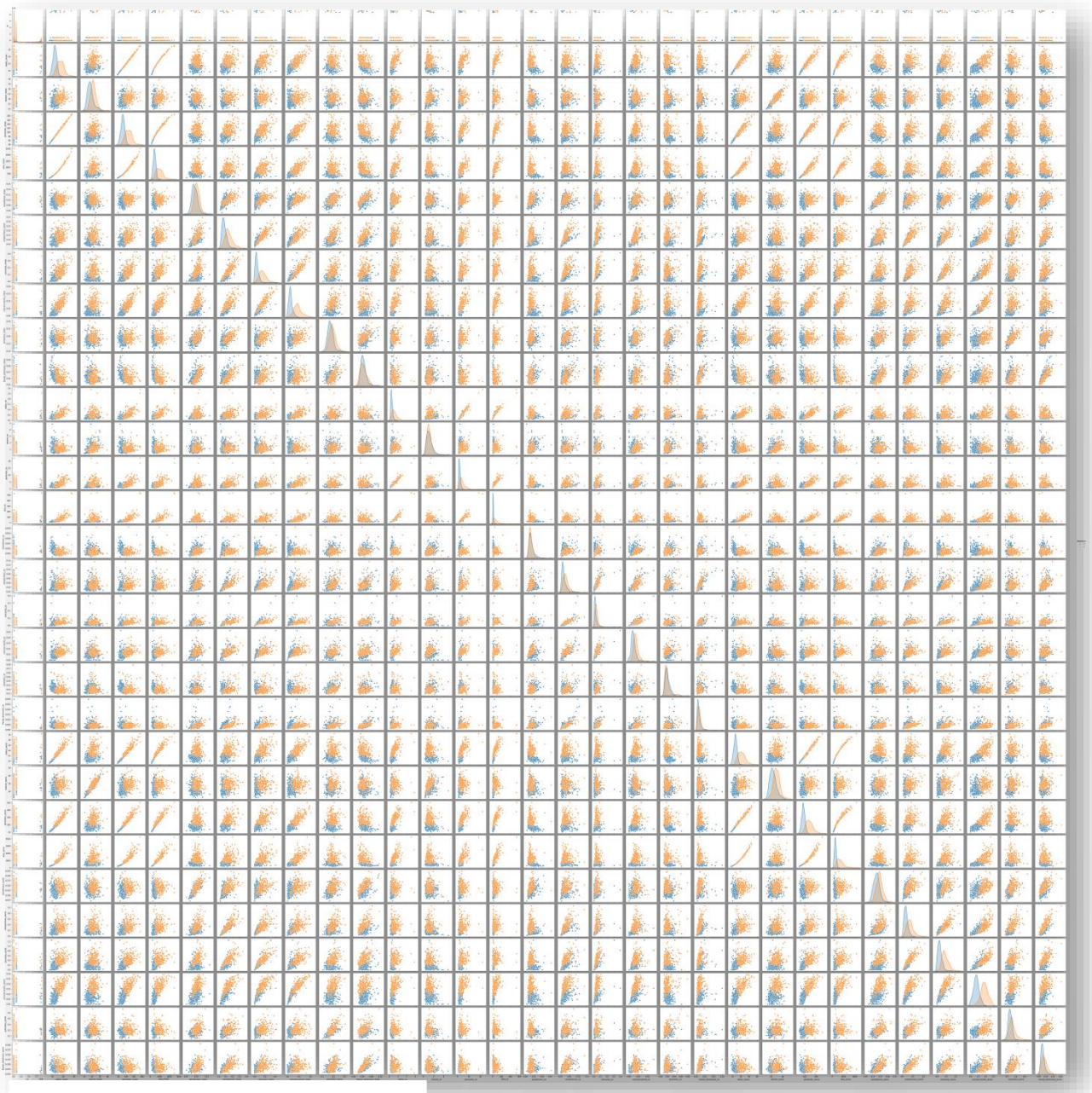

Figure 11.2 Pair Plot

The pair plot of all columns that show the diagnostic points is shown in Figure 11.2. The orange points is for 1 and blue is for 0. Basically, the pair is used to show the numeric distribution in the scatter plot.

3.3. Heat Map

To visualize the data further, print the first five rows of dataset. In python this is done by using pandas **head** function.

```
In [12]: df.head(5)
```

Out[12]:

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

5 rows × 32 columns

Figure 12 First five rows of DataFrame

Following this correlation of columns is performed by using pandas **corr** function.

```
In [13]: #Get the correlation of the columns
         df.corr().head()
```

Out[13]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean
id	1.000000	0.039769	0.074626	0.099770	0.073159	0.096893	-0.012968	0.000096	0.050080
diagnosis	0.039769	1.000000	0.730029	0.415185	0.742636	0.708984	0.358560	0.596534	0.696360
radius_mean	0.074626	0.730029	1.000000	0.323782	0.997855	0.987357	0.170581	0.506124	0.676764
texture_mean	0.099770	0.415185	0.323782	1.000000	0.329533	0.321086	-0.023389	0.236702	0.302418
perimeter_mean	0.073159	0.742636	0.997855	0.329533	1.000000	0.986507	0.207278	0.556936	0.716136

5 rows × 32 columns

Figure 13 Correlation of columns

From figure 13, correlation analysis of columns in reference to pair plot is displayed which is helpful to analyse the relationship among the features in individual patients.

The visualization of correlation is more reliable and easier via heatmap. A heatmap is a two dimentional illustration of data by colors. The use of colors makes the visualization better as it might be tough to grasp a data if presented numerically. Heatmap provides a direct visual outline of data. So, to find the correlation between each feature and diagnosis heatmap is vizualized using correlation matrix.

In python heatmap is formed by using seaborn **heatmap** function.

```
In [14]: plt.figure(figsize=(20,20))
         sns.heatmap(df.corr(), annot=True, fmt='.0%')

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x1fd76e7ec88>
```

Figure 14.1 Function to make Heat Map

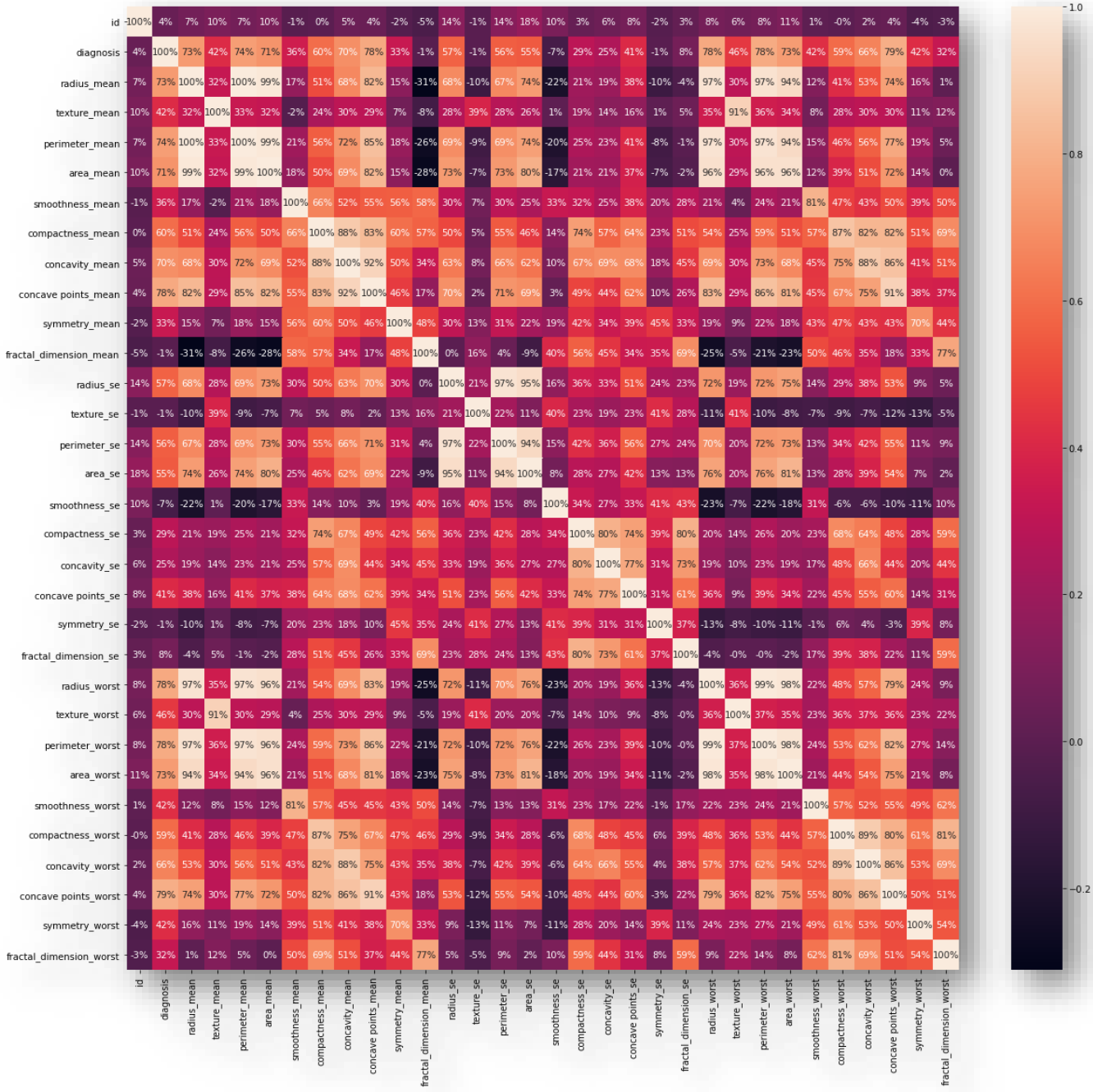

Figure 14.2 Heat Map

Looking at the heatmap, the focus is on the light and the dark areas. It shows the strength of correlation. The light area represents high correlation while dark area represents weak correlation among the attributes/characteristics. The added annotation in heatmap, which are the actual correlation values, enables to easily form a conclusion.

4. Model selection

Model selection is the process of choosing different machine learning algorithms. More than one kind of machine learning techniques can be used. The algorithms are classified in two major groups:

1. **Supervised learning**: Method in which the machine is trained on data which labels the input and output. The model will learn the training data to predict outcomes and process future data. It is again divided into two groups viz.
   Regression: It is used when the result is continuous or real.
   Classification: It is used when the result is a category.

2. **Unsupervised learning**: Method that trains the computer on data that marks input and output. The model must learn to predict effects from the training data and process future data.

The dependent variable has two set of values in our breast cancer dataset, either malignant (M) or Benign (B). So, the supervised learning classification algorithm is used. The different types of classification algorithm used in this model are:

- Logistic Regression
- K Neighbour classifier
- SVC linear
- Gaussian Naïve Bayes
- Decision Tree classifier

The data is set up for model as per following:

4.1. Split data set

The first division is the splitting of data into a collection of characteristics, also known as an independent data set (X), and a set of target data also known as the dependent data set (Y).

In python it is done by using **slice** operation. In X variable, values of all rows and column from 3 -31 is assigned. And in Y variable, all rows but only second column is selected and assigned.

```
In [15]: X = df.iloc[:, 2:31].values
         Y = df.iloc[:, 1].values
```

Figure 15 Function to split independent and dependent variable

Now, split the data again but this time into 75% training and 25% research data sets. Training data is used in machine learning to match the model and test data to test for it. The developed models are for the prediction of uncertain outcomes, i.e. test set. Therefore the dataset is divided into train and test set to verify precision and accuracy by training and testing it on it.
It was done in python, using sklearn.model_selection train_test_split.

```
In [16]: 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)
```

Figure 16 Splitting of 75% training set and 25% test set from DataFrame

4.2. Feature scaling

Features in the dataset often differ in magnitude, units, and range. But since most machine learning algorithms use eucledian distance between two data points, with the aid of scaling it is necessary to bring all features to the same magnitude level. It will result in a different range of the feature / independent data. 0-100, or 0-1, for example.
In python this is done by using **StandardScaler** function from **sklearn.preprocessing**.

```
In [17]: #Feature Scaling
         from sklearn.preprocessing import StandardScaler
         sc = StandardScaler()
         X_train = sc.fit_transform(X_train)
         X_test = sc.transform(X_test)
```

Figure 17 Feature scaling

4.3. Create function to hold different models

Function is created to hold all the models used in dataset to make classification. For python the sklearn library uses the algorithm to import all the classification algorithm methods.
Upon importing the library, and then using the Logistic Regression Algorithm to the Training Set, K Neighbors Classifier System of neighbors to use Nearest Neighbor algorithm, SVC linear svm class method to use Vector Machine Algorithm Help, GaussianNB naïve bayes class method to use Naïve Bayes Algorithm, and Decision Tree node classifier to use Decision Tree Algorithm. Exactness of each model on the training data is also printed within this feature.

```
In [27]: def models(X_train,Y_train):
           #Using Logistic Regression
           from sklearn.linear_model import LogisticRegression
           log = LogisticRegression(random_state = 0)
           log.fit(X_train, Y_train)
           #Using KNeighborsClassifier
           from sklearn.neighbors import KNeighborsClassifier
           knn = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
           knn.fit(X_train, Y_train)
           #Using SVC linear
           from sklearn.svm import SVC
           svc_lin = SVC(kernel = 'linear', random_state = 0)
           svc_lin.fit(X_train, Y_train)
           #Using GaussianNB
           from sklearn.naive_bayes import GaussianNB
           gauss = GaussianNB()
           gauss.fit(X_train, Y_train)
           #Using DecisionTreeClassifier
           from sklearn.tree import DecisionTreeClassifier
           tree = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
           tree.fit(X_train, Y_train)
           #Using RandomForestClassifier method of ensemble class to use Random Forest Classification algorithm
           from sklearn.ensemble import RandomForestClassifier
           #print model accuracy on the training data.
           print('[0]Logistic Regression Training Accuracy:', log.score(X_train, Y_train))
           print('[1]K Nearest Neighbor Training Accuracy:', knn.score(X_train, Y_train))
           print('[2]Support Vector Machine (Linear Classifier) Training Accuracy:', svc_lin.score(X_train, Y_train))
           print('[3]Gaussian Naive Bayes Training Accuracy:', gauss.score(X_train, Y_train))
           print('[4]Decision Tree Classifier Training Accuracy:', tree.score(X_train, Y_train))
           return log, knn, svc_lin, gauss, tree
```

Figure 18 Function to build models and its accuracies

4.4. Create model

Build a database that includes all the models, and analyze the accuracy score for each database on the training results. It is done to figure out and determine whether or not the patient has cancer.

```
In [28]: model = models(X_train,Y_train)

[0]Logistic Regression Training Accuracy: 0.9906103286384976
[1]K Nearest Neighbor Training Accuracy: 0.9765258215962441
[2]Support Vector Machine (Linear Classifier) Training Accuracy: 0.9882629107981221
[3]Gaussian Naive Bayes Training Accuracy: 0.9507042253521126
[4]Decision Tree Classifier Training Accuracy: 1.0
```

Figure 19 Accuracies of training set model

Precise validation of the data is used to check the model. To test for accuracy, import metric class confusion matrix method. It summarizes how a classification algorithm works. The confusion matrix calculates and provides what model of classification is getting right and what kinds of errors it makes. The numbers of correct and incorrect predictions are summarized with count values, and each class breaks down. It provides insights into the error made by the classifier, and the types of error made.

```
In [29]: from sklearn.metrics import confusion_matrix
         for i in range(len(model)):
           cm = confusion_matrix(Y_test, model[i].predict(X_test))

           TN = cm[0][0]
           TP = cm[1][1]
           FN = cm[1][0]
           FP = cm[0][1]

           print(cm)
           print('Model[{}] Testing Accuracy = "{}!"'.format(i,  (TP + TN) / (TP + TN + FN + FP)))
           print()# Print a new line

[[86  4]
 [ 4 49]]
Model[0] Testing Accuracy = "0.9440559440559441!"

[[89  1]
 [ 5 48]]
Model[1] Testing Accuracy = "0.958041958041958!"

[[87  3]
 [ 2 51]]
Model[2] Testing Accuracy = "0.965034965034965!"

[[85  5]
 [ 6 47]]
Model[3] Testing Accuracy = "0.9230769230769231!"

[[84  6]
 [ 1 52]]
Model[4] Testing Accuracy = "0.951048951048951!"
```

Figure 20 Confusion matrix of test set

Confusion matrix accuracy of each matrix is displayed. Also, here in case of breast cancer, **type II** error is more dangerous error and would cause greater consequence. Thus, considering both type I and II errors in each model with respect to its accuracies, model 2 i.e. SVM linear is comparatively more accurate than other mentioned model for breast cancer detection.

4.5. Cross validation

Since our model does not have huge data set, testing set is left with few observations to lead any real conclusion. So, cross validation is performed on the data set to make predictions on all data. Also, to check if the machine learning model is over fitted,

generalized or under fitted, cross validation of model is performed. If the model is in fit, cross validation will have a high training and test error and if the model is over-fit, cross validation will have an extremely low training error but a high test error. Over fitting and under fitting is the common error that occurs in selection of a model. Thus, to overcome this, cross validation of 10 fold is performed to validate the result.

```
In [30]: from sklearn.model_selection import cross_val_score
         cross_validation0 = cross_val_score(model[0],  X= X_train, y = Y_train, cv= 10)
         cross_validation1 = cross_val_score(model[1],  X= X_train, y = Y_train, cv= 10)
         cross_validation2 = cross_val_score(model[2],  X= X_train, y = Y_train, cv= 10)
         cross_validation3 = cross_val_score(model[3],  X= X_train, y = Y_train, cv= 10)
         cross_validation4 = cross_val_score(model[4],  X= X_train, y = Y_train, cv= 10)

         cross_validation0.mean(), cross_validation1.mean(), cross_validation2.mean(), cross_validation3.mean(),cross_validation4.mean()

Out[30]: (0.9860465116279069,
          0.9648394241417497,
          0.9719269102990034,
          0.9437984496124031,
          0.9270764119601329)
```

Figure 21 Cross validation

5. Result and Discussion

The result of cross validation of each model is compared against training and testing set. Taking confusion matrix into consideration and analysing the accuracies, it is observed that although, SVM linear model is slightly imbalanced with cross validation 97.19%, training set 98.83% and testing 96.50%, but this can be generalized by changing and modifying the training and testing set. The number of observations in the training and test collection also impacts the quality of the data. Thus, with performance metric of **97.19%**, SVM linear is comparatively more accurate than Logistic Regression, KNN, GaussianNB, and Decision Tree in detecting breast cancer.

Accuracy check of SVC linear model with actual values

As SVC linear is comparatively more accurate for detection of breast cancer, the accuracy of model is analysed with the actual values of breast cancer diagnosis. This will help in highlighting the result of detection with percentage of error more clearly.

```
In [22]: #Print Prediction of SVC linear model
pred = model[2].predict(X_test)
print(pred)

#Print a space
print()

#Print the actual values
print(Y_test)
```

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

[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]
```

Figure 22 Accuracy check

From this accuracy check it can be analysed that, the model SVC linear with performance metric of approximately 97.19% can predict and diagnose if a patient has cancer or not.

6. Conclusion

This study attempts to analyse various supervised machine learning algorithms and select the most accurate model in detection of breast cancer. The work focused in advancement of predictive models with the help of python to achieve better accuracy in predicting correct outcomes. The analysis of result signifies that, integration of data, feature scaling along with different classification method and analysis provide markedly successful tool in prediction. It has also observed that the model misdiagnosed few patients with cancer when they were not having cancer and vice versa. Although, the model is accurate but when dealing with lives of people, further research in building the most accurate and precise model must be carried out for better performance of classification techniques and get the accuracy as close to 100% as possible. Thus, the tuning of each of the models is necessary with the building of more reliable model.

4. References

1. http:// www.imaginis.com/breasthealth/breast_cancer.asp, Last Accessed August 2007.
2. West, D., Mangiameli, P., Rampal, R., & West, V. (2005). Ensemble strategies for a medical diagnosis decision support system: A breast cancer diagnosis application. European Journal of Operational Research(162), 532–551
3. https://www.mayoclinic.org/diseases-conditions/breast-cancer/symptoms-causes/syc-20352470
4. Pavlopoulos, S.A.; Delopoulos, A.N. Designing and implementing the transition to a fully digital hospital. IEEE Trans. Inf. Technol. Biomed. 1999, 3, 6–19.
5. Barracliffe, L.; Arandjelović, O.; Humphris, G. A pilot study of breast cancer patients: Can machine learning predict healthcare professionals' responses to patient emotions? In Proceedings of the International Conference on Bioinformatics and Computational Biology, Honolulu, HI, USA, 20–22 March 2017; pp. 101–106.
6. Birkett, C.; Arandjelović, O.; Humphris, G. Towards objective and reproducible study of patient-doctor interaction: Automatic text analysis based VR-CoDES annotation of consultation transcripts. In Proceedings of the IEEE Engineering in Medicine and Biology Society Conference, Jeju Island, Korea, 11–15 July 2017; pp. 2638–2641.
7. A. J. Cruz and D. S. Wishart, "Applications of machine learning in cancer prediction and prognosis," Cancer Informatics, vol. 2, pp. 59–77, 2006.
8. G. Valvano, G. Santini, N. Martini et al., "Convolutional neural networks for the segmentation of microcalcification in mammography imaging," Journal of Healthcare Engineering, vol. 2019, Article ID 9360941, 9 pages, 2019.
9. M. F. Akay, "Support vector machines combined with feature selection for breast cancer diagnosis," Expert Systems with Applications, vol. 36, no. 2, pp. 3240–3247, 2009.
10. P. Salembier and L. Garrido, "Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval," IEEE Transactions on Image Processing, vol. 9, no. 4, pp. 561–576, 2000.
11. Mangasarian, O.L.; Setiono, R.; Wolberg, W.H. Pattern recognition via linear programming: Theory and application to medical diagnosis. In Large-Scale Numerical Optimization; SIAM: Philadelphia, PA, USA, 1990; pp. 22–31.
12. Wolberg, W.H.; Mangasarian, O.L. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. Proc. Natl. Acad. Sci. USA 1990, 87, 9193–9196.

13. Sharma, A.; Kulshrestha, S.; Daniel, S. Machine learning approaches for breast cancer diagnosis and prognosis. In Proceedings of the International Conference on Soft Computing and Its Engineering Applications, Changa, India, 1–2 December 2017.

14. https://www.kaggle.com/uciml/breast-cancer-wisconsin-data

5. Annexure

1. The benchmark data of interest for detection of breast cancer can be referred from https://www.kaggle.com/uciml/breast-cancer-wisconsin-data

2. For more reference, the program of this paper is uploaded in GitHub link https://github.com/neha-2568/Detection-of-Breast-Cancer

3. The more clear vision of Pair Plot and Heat Map is available in GitHub link https://github.com/neha-2568/Detection-of-Breast-Cancer