CS3120 Machine Learning
and
Neural Computing

Breast Cancer Detection Using Random Forest Machine Learning technique

[Assignment 2]

Name: Umeshika Dissanayaka

Index Number: s14320

Date: 16/10/2021

TABLE OF CONTENTS

1 INTRODUCTION

In the current world, Breast cancer is the most well-known harm among ladies around the world. There is a main connection between parameters that can be gathered in routine blood analysis and breast cancer risk yet and also few questions stay about the job of adipokines. Therefore, this assignment is expected to decide the connection between weight record (BMI), glycaemia, insulinemia, insulin-obstruction, blood adipokine levels and growth qualities in a Portuguese gathering of pre-and postmenopausal overweight/stout ladies with Breast cancer.

1.1 About dataset

This dataset was observed and measured by 'BMC Cancer' within 166 participants of Clinical features that were obtained for 64 patients who had breast cancer and 52 healthy controls. Therefore, this data set mainly included several clinical features which can be gathered in routine blood analysis such as age, BMI, Glucose, Insulin, HOMA, Leptin, Adiponectin, Resistin and MCP-1. These features can be defined as all quantitative 10 predictors that are binary dependent variables that represent patients who had breast cancer or not. In addition, this data set is implemented the Machine learning algorithms that are logistic regression, support vector machines and random forests. Furthermore, this data set is accepted for Monte Carlo Cross-Validation with 95% confidence intervals for the sensitivity.

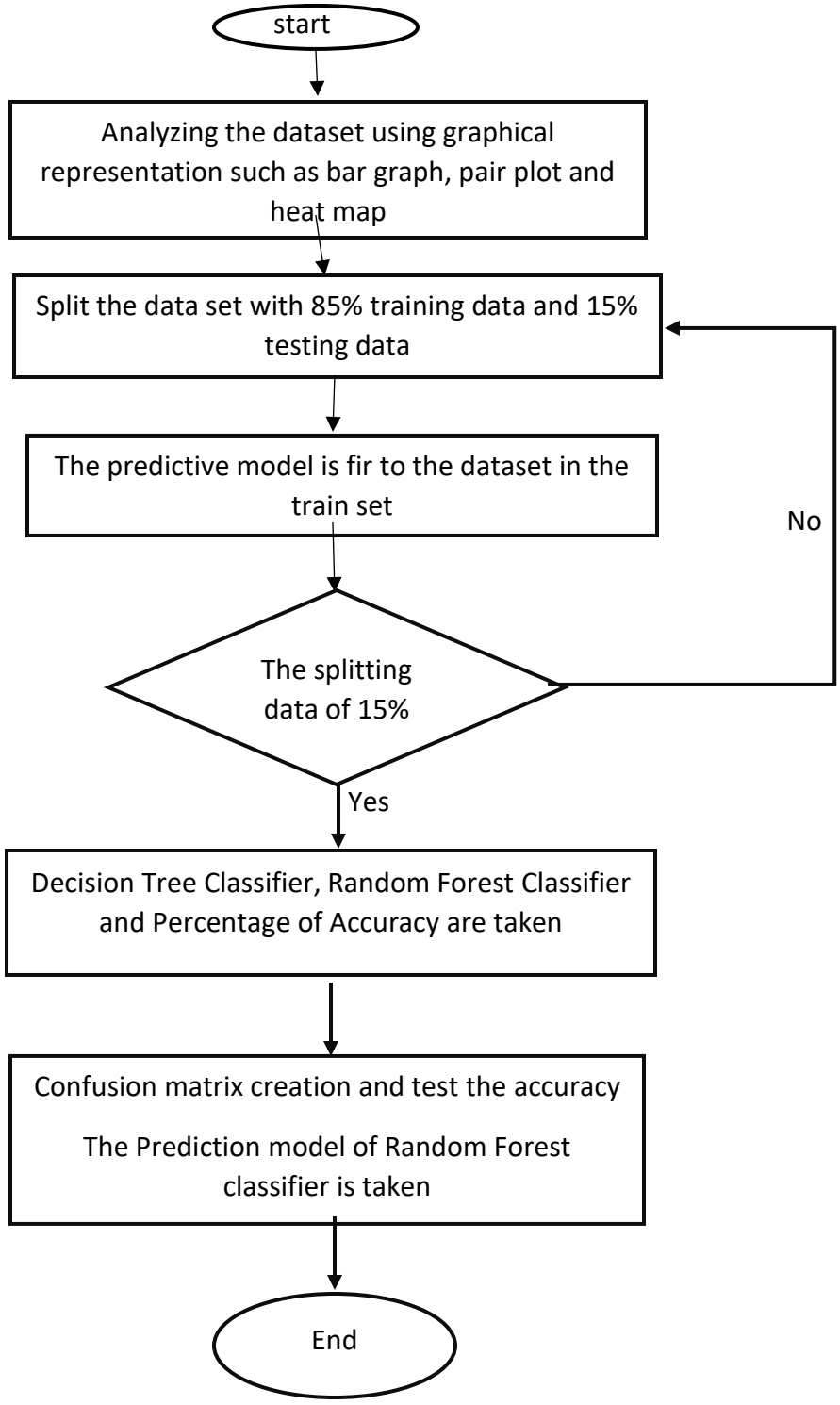
Attribute Information:

- Age (years)
- BMI (kg/m2)
- Glucose (mg/dL)
- Insulin (µU/mL)
- HOMA Leptin (ng/mL)
- Adiponectin (µg/mL)
- Resistin (ng/mL)
- MCP-1(pg/dL)
- Classification Labels: 1=Healthy controls 2=Patients

1.2 Goal

To Predict a new model using Random Forest Classifier of Machine Learning technique to detect the Breast Cancer patient or healthy patient. Therefore, anthropometric data and parameters which can be gathered in routine blood analysis are defined as independent variables and patient or healthy person are defined as the dependent variable.

1.3 The approach



This is the approach to predict a person is having breast cancer or not.

2 METHODOLOGY

- The libraries were imported

Import Libraries

```
[ ]  #import libraries
     import numpy as np
     import pandas as pd
     from sklearn import preprocessing
     import matplotlib.pyplot as plt
     import seaborn as sns
     import warnings
     warnings.simplefilter(action='ignore')
```

- the dataset was imported.

Import Dataset

```
[ ]  import pandas as pd
     df = pd.read_excel('https://archive.ics.uci.edu/ml/machine-learning-databases/00451/dataR2.xlsx')
     df.head()
```

- The Number of rows and columns were counted in the data set and Counted the empty then the missing value was dropped.

```
#Number of rows and columns are counted in the data set
df.shape
```

```
(116, 10)
```

```
#Count the empty and The missing value was dropped
df.isna().sum()
```

- The count of the number of Healthy controls & Patients is taken by using the 'value_counts()' command.
- The count for visualized Healthy controls & Patients were visualized by using a bar graph

```
#A count of the number of Healthy controls  & Patients are taken by using value_counts()
df['Classification'].value_counts()
```

```
2    64
1    52
Name: Classification, dtype: int64
```

```
#The count is Visualized using bar graph
sns.countplot(df['Classification'],label="Count")
```

- After that, the data types were checked and a pair plot was created.

```
[ ]  #The data types are checked
     df.dtypes
```

```
sns.pairplot(df, hue="Classification")
```

- The correlation of the columns was taken then it was visualized by using creating a heat map.

```
#The correlation of the columns are taken
df.corr()
```

```
plt.figure(figsize=(12,12))
sns.heatmap(df.corr(), cbar=True, annot=True, cmap="Reds", fmt='.0%')
```

- The independent variables and dependent variables were defined and split the dataset into a training set and test set concerning 85% training data and 15% testing data.
- the model was trained on the training set with Standard Feature Scaling

Define X and Y

```
[ ]  X = df.drop('Classification',axis=1).values
     Y = df['Classification'].values
```

Split the dataset in training set and test set

85% training data and 15% testing data

```
[ ]  from sklearn.model_selection import train_test_split
     X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.15, random_state = 0)
```

Train the model on the training set with Standard Feature Scaling

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

- The special function was defined to hold Two different models of Random Forest Classification algorithm and Decision Tree Classifier which was named "classification_function".

```
def classification_function(X_train,Y_train):

  #The DecisionTreeClassifier is calculated
  from sklearn.tree import DecisionTreeClassifier
  tree = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
  tree.fit(X_train, Y_train)

  #The Random Forest Classification algorithm is calculated by using RandomForestClassifier method
  from sklearn.ensemble import RandomForestClassifier
  forest = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 0)
  forest.fit(X_train, Y_train)

  #The model accuracy on the training data are taken
  print('[0]Decision Tree Classifier Training Percentage of Accuracy:', tree.score(X_train, Y_train)*100,'%')
  print('[1]Random Forest Classifier Training Percentage of Accuracy:', forest.score(X_train, Y_train)*100,'%')

  return tree, forest
```

- The Percentage of Accuracy was taken for trained data.

```
Resluts = classification_function(X_train,Y_train)
```

```
[0]Decision Tree Classifier Training Percentage of Accuracy: 100.0 %
[1]Random Forest Classifier Training Percentage of Accuracy: 97.95918367346938 %
```

- The confusion matrix was created, visualized and test accurately.

Class		Actual	
		Positive	**Negative**
Predict	**Positive**	True Positive	False Negative
	Negative	False Positive	True Negative

Figure 2-1 The confusion matrix for the accuracy of the models on the test data. (reference:https://scholar.ui.ac.id/en/publications/feature-selection-using-random-forest-classifier-for-predicting-p)

Where TN is called the true negative rate, FP is called the False Positive rate, TP is called the True Positive rate and FN is called the False Negative rate of patients with or without cancer that was misdiagnosed as not having cancer.

Using these 4 values TN, TP, FN, FP the accuracy, Precision and recall equation can be defined as follows.

$$Accuracy = \frac{The\ number\ of\ correctly\ predictions}{The\ total\ number\ of\ the\ data}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

```
from sklearn.metrics import confusion_matrix
for i in range(len(Resluts)):

  cm = confusion_matrix(Y_test, Resluts[i].predict(X_test))

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

  plt.figure(figsize=(5,4))
  sns.heatmap(cm,annot=True)
  plt.xlabel('Predicted')
  plt.ylabel('Truth')
  plt.title("Confusion matrix for model")


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

- Using libraries the results of accuracy, precision & recall were obtained again.

```
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score

for i in range(len(Resluts)):
  print('Model ',i)
  #Check precision, recall, f1-score
  print( classification_report(Y_test, Resluts[i].predict(X_test)) )
  #Another way to get the models accuracy on the test data
  print( accuracy_score(Y_test, Resluts[i].predict(X_test)))
  print()#Print a new line
```

- Finally, the Prediction of Random Forest Classifier model was taken and compared with real values.

```
#The Prediction of Random Forest Classifier model
pred = Resluts[1].predict(X_test)
print(pred)

#Print a Blank
print()

#The Real values
print(Y_test)
```

3 RESULTS AND ANALYSIS

	Age	BMI	Glucose	Insulin	HOMA	Leptin	Adiponectin	Resistin	MCP.1	Classification
0	48	23.500000	70	2.707	0.467409	8.8071	9.702400	7.99585	417.114	1
1	83	20.690495	92	3.115	0.706897	8.8438	5.429285	4.06405	468.786	1
2	82	23.124670	91	4.498	1.009651	17.9393	22.432040	9.27715	554.697	1
3	68	21.367521	77	3.226	0.612725	9.8827	7.169560	12.76600	928.220	1
4	86	21.111111	92	3.549	0.805386	6.6994	4.819240	10.57635	773.920	1

This is the data set was taken to analyze and classification attribute has two label 1 that represented healthy controls and 2 represented patient.

```
df.shape
```

```
(116, 10)
```

```
#Count the empty and The missing value was dropped
df.isna().sum()
```

```
Age               0
BMI               0
Glucose           0
Insulin           0
HOMA              0
Leptin            0
Adiponectin       0
Resistin          0
MCP.1             0
Classification    0
dtype: int64
```

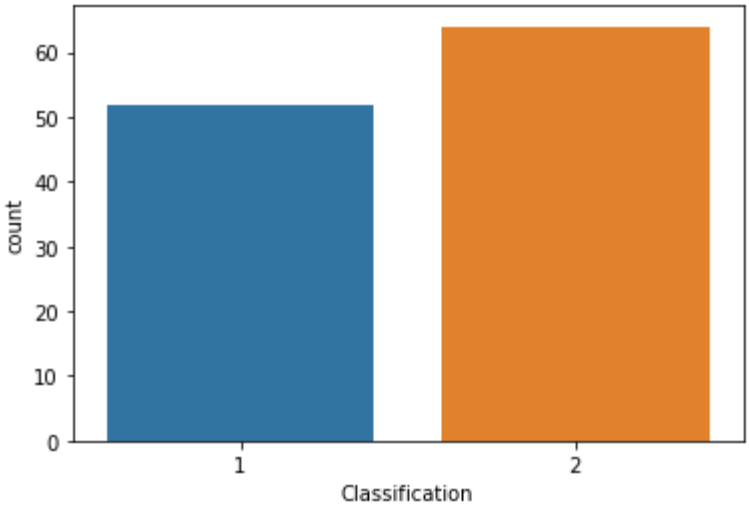
And also in this dataset has 116 rows and 10 columns. There were not included any missing values and all data types are “int64”.

```
df['Classification'].value_counts()
```

```
2    64
1    52
Name: Classification, dtype: int64
```

```
#The count is Visualized using bar graph
sns.countplot(df['Classification'],label="Count")
```

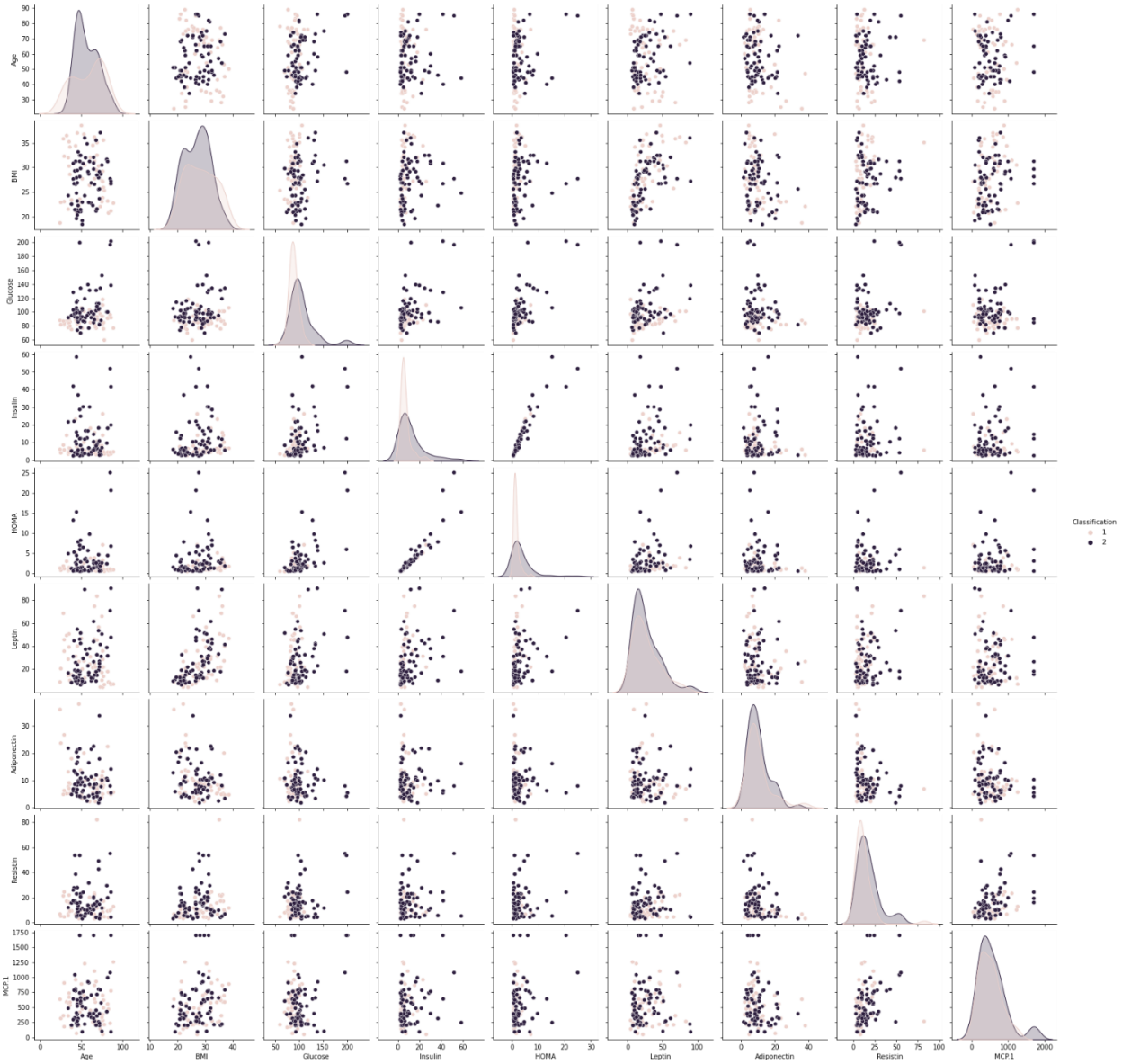
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe72f56d590>
```



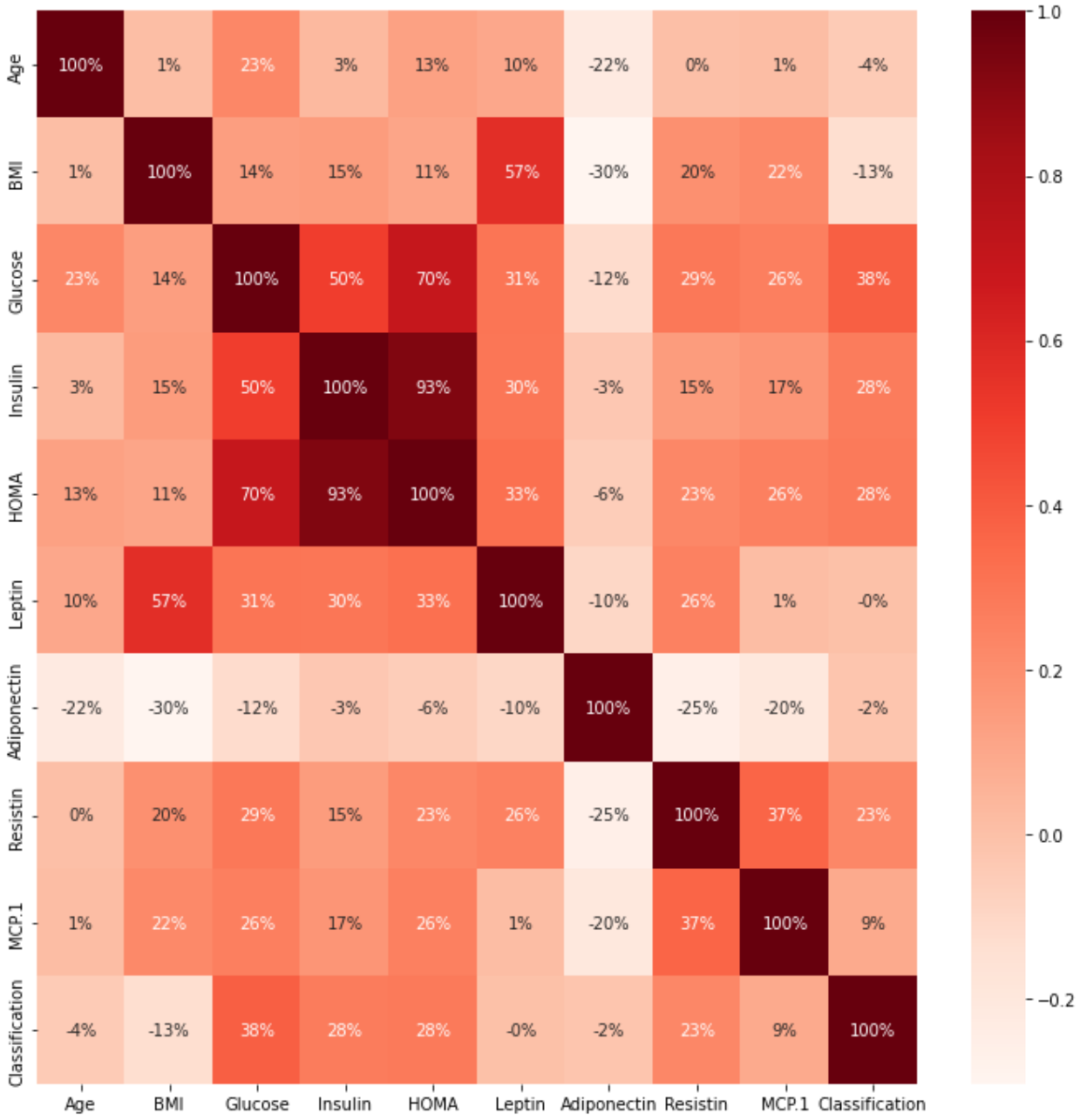
This bar graph represents the classification of healthy controls and patients. In addition, this data set had included 64 patients and 52 healthy controls. Therefore these two categorical data counts are almost equal. Hence these bars do not need to be transformed.

	Age	BMI	Glucose	Insulin	HOMA	Leptin	Adiponectin	Resistin	MCP.1	Classification
Age	1.000000	0.008530	0.230106	0.032495	0.127033	0.102626	-0.219813	0.002742	0.013462	-0.043555
BMI	0.008530	1.000000	0.138845	0.145295	0.114480	0.569593	-0.302735	0.195350	0.224038	-0.132586
Glucose	0.230106	0.138845	1.000000	0.504653	0.696212	0.305080	-0.122121	0.291327	0.264879	0.384315
Insulin	0.032495	0.145295	0.504653	1.000000	0.932198	0.301462	-0.031296	0.146731	0.174356	0.276804
HOMA	0.127033	0.114480	0.696212	0.932198	1.000000	0.327210	-0.056337	0.231101	0.259529	0.284012
Leptin	0.102626	0.569593	0.305080	0.301462	0.327210	1.000000	-0.095389	0.256234	0.014009	-0.001078
Adiponectin	-0.219813	-0.302735	-0.122121	-0.031296	-0.056337	-0.095389	1.000000	-0.252363	-0.200694	-0.019490
Resistin	0.002742	0.195350	0.291327	0.146731	0.231101	0.256234	-0.252363	1.000000	0.366474	0.227310
MCP.1	0.013462	0.224038	0.264879	0.174356	0.259529	0.014009	-0.200694	0.366474	1.000000	0.091381
Classification	-0.043555	-0.132586	0.384315	0.276804	0.284012	-0.001078	-0.019490	0.227310	0.091381	1.000000

This is the correlation of the datasets, and it is involved both positive and negative correlations.



This figure panel shows the pair plot of each attribute and which have the scatter plots



This figure has visualized the correlation by using a heat map. Therefore, the relationship percentage between the two attributes can be effortlessly obtained using this heat map. For example, if it needs to take the percentage of the relationship between the MCP1 and BMI then using this heat map can be obtained the commonplace square which has 22%.

```
[0]Decision Tree Classifier Training Percentage of Accuracy: 100.0 %
[1]Random Forest Classifier Training Percentage of Accuracy: 97.95918367346938 %
```

The data set split 85% training data and 15% testing data hence the special defined function was given the training accuracy percentage as 100% for the Decision tree classifier and 97.96% for the Random Forest classifier. Thus, both result it was concluded that Decision tree classifier model with 15% test data are more accurate than Random forest classifier.

The confusion matrix is represented how many patients each model have not breast cancer.

False Positive (FP) = A test outcome that inaccurately shows that a specific condition or quality is available.

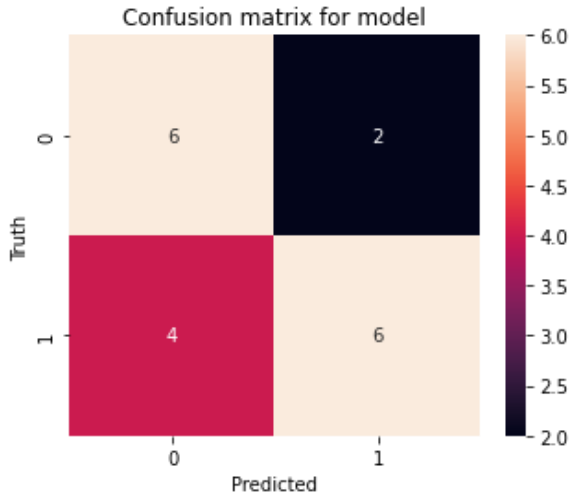
True Positive (TP) = A test outcome that accurately shows that a specific condition or quality is available and real up-sides that are accurately recognized thusly.

True Negative (TN) = Specificity is the genuine negative rate of the extent of real negatives that are effectively distinguished all things considered.

False Negative (FN) =test outcome that shows an individual doesn't have cancer when the individual has it. A test outcome that demonstrates that a condition doesn't hold, while indeed it does

		Prediction	
		0	1
Actual	0	TN	FP
	1	FN	TP

Decision tree classifier




```
[[6 2]
 [4 6]]
Model[0] Testing Accuracy  = 0.6666666
Model[0] Testing Precision = 0.75
Model[0] Testing Recall    = 0.6
```

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

$$Accuracy = \frac{6 + 6}{6 + 6 + 4 + 2} = 0.666667$$

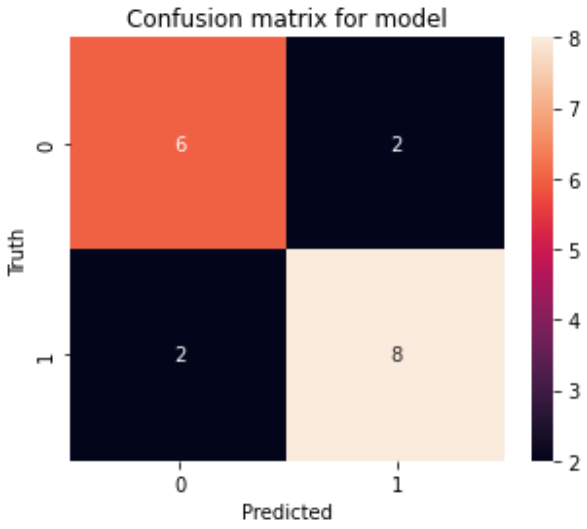
$$Precision = \frac{TP}{TP + FP}$$

$$Precision = \frac{6}{6 + 2} = 0.75$$

$$Recall = \frac{TP}{TP + FN}$$

$$Recall = \frac{6}{6 + 4} = 0.6$$

Random Forest classifier




```
[[6 2]
 [2 8]]
Model[1] Testing Accuracy  = 0.77777777
Model[1] Testing Precision = 0.8
Model[1] Testing Recall    = 0.8
```

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

$$Accuracy = \frac{6 + 8}{6 + 8 + 2 + 2} = 0.7777778$$

$$Precision = \frac{TP}{TP + FP}$$

$$Precision = \frac{8}{8 + 2} = 0.8$$

$$Recall = \frac{TP}{TP + FN}$$

$$Recall = \frac{8}{8 + 2} = 0.8$$

When comparing the Decision tree classifier and Random Forest classifier confusion matrix it was concluded that the most accurate could be obtained using the Random Forest classifier for this data set.

```
              precision    recall  f1-score   support

           1       0.60      0.75      0.67         8
           2       0.75      0.60      0.67        10

    accuracy                           0.67        18
   macro avg       0.68      0.68      0.67        18
weighted avg       0.68      0.67      0.67        18

0.6666666666666666

Model  1
              precision    recall  f1-score   support

           1       0.75      0.75      0.75         8
           2       0.80      0.80      0.80        10

    accuracy                           0.78        18
   macro avg       0.78      0.78      0.78        18
weighted avg       0.78      0.78      0.78        18

0.7777777777777778
```

The same results of precision, accuracy and recall were obtained using the libraries

4 INFERENCES AND CONCLUSIONS

```
#The Prediction of Random Forest Classifier model
pred = Resluts[1].predict(X_test)
print(pred)

#Print a Blank
print()

#The actual values
print(Y_test)
```

```
[1 2 1 1 2 2 2 2 1 2 1 1 2 1 2 2 1 2]

[1 2 2 1 1 2 2 2 2 2 1 1 2 1 1 2 1 2]
```

When comparing prediction values and test values, two-person who have not been diagnosed but consider two of the patients as having breast cancer growth and it was given two patients that had already have cancer as healthy controls. Somehow this model is acceptable when managing the existence of other data, therefore this model is to be better and almost as near 100% as could more tuning of essential.

According to this exactness and measurements above the model that visualized out precious on the train, information was the Random Forest Classifier with an accuracy percentage of about 97.95918367346938 % and test information was the Random Forest Classifier with an accuracy percentage of about 77.77777777 %. Therefore, this model has identified disease cells in patients much accurate way. In conclusion, this data set can be accepted for the Random Forest Classifier model.

5 SOURCE CODE

https://colab.research.google.com/drive/1r-0AeN-ZPMFDJuDbBbwFe0igPfsoTf2G?usp=sharing

6 REFERENCES

[1] *Huljanah, M., Rustam, Z., Utama, S. and Siswantining, T., 2019. Feature Selection using Random Forest Classifier for Predicting Prostate Cancer. IOP Conference Series: Materials Science and Engineering, 546, p.052031.*

[2] *Medium. 2021. Breast Cancer Detection Using Machine Learning. [online] Available at: < https://randerson112358.medium.com/breast-cancer-detection-using-machine-learning-38820fe98982> [Accessed 16 October 2021].*

[3] *Crisóstomo, J., Matafome, P., Santos-Silva, D., Gomes, A., Gomes, M., Patrício, M., Letra, L., Sarmento-Ribeiro, A., Santos, L. and Seiça, R., 2016. Hyperresistinemia and metabolic dysregulation: a risky crosstalk in obese breast cancer. Endocrine, 53(2), pp.433-442.*

[4] *Patrício, M., Pereira, J., Crisóstomo, J., Matafome, P., Gomes, M., Seiça, R. and Caramelo, F., 2018. Using Resistin, glucose, age and BMI to predict the presence of breast cancer. BMC Cancer, 18(1).*

[5] *Archive.ics.uci.edu. 2021. UCI Machine Learning Repository: Breast Cancer Coimbra Data Set. [online] Available at: <https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Coimbra# > [Accessed 16 October 2021]*

[6] *Analytics Vidhya. 2021. Random Forest | Introduction to Random Forest Algorithm. [online] Available at: <https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/ > [Accessed 16 October 2021]..*