Turkish Aeronautical Association University

Degree in Computer Engineering

1st Semester 2022-2023

Practical Work of Data Analysis Course - CENG463

Work done by:

Marya Aktaş

200444001

Contents

Introduction

In the practical work of 'Data Analysis' course, a code written in Matlab that performs some operations on a chosen dataset was requested. This dataset should be numeric and can be used for classification.

For my practical work, I chose 'Breast Cancer Coimbra' dataset from https://archive.ics.uci.edu/ml/datasets.php. This data set keeps the records of clinical features that are observed or measured for 64 patients with breast cancer and 52 healthy controls in 2018. There are no missing values, all attributes are numeric, and it can be used for classification. In this data set, there are 10 predictors, all quantitative, and a binary dependent variable, indicating the presence or absence of breast cancer. The predictors are anthropometric data and parameters which can be gathered in routine blood analysis. The attributes of this data set are:

Quantitative Attributes:
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)

Labels(Classification Attribute):
1=Healthy controls
2=Patients

The first 5 instance of this data set is as follows:

116×10 table

Age	BMI	Glucose	Insulin	HOMA	Leptin	Adiponectin	Resistin	MCP_1	Classification
48	23.5	70	2.707	0.46741	8.8071	9.7024	7.9958	417.11	1
83	20.69	92	3.115	0.7069	8.8438	5.4293	4.064	468.79	1
82	23.125	91	4.498	1.0097	17.939	22.432	9.2772	554.7	1
68	21.368	77	3.226	0.61272	9.8827	7.1696	12.766	928.22	1
86	21.111	92	3.549	0.80539	6.6994	4.8192	10.576	773.92	1

(Figure 1)

The code must perform:

- Reading the chosen data set
- Finding the centered data matrix of this data set
- Finding mean, median, sum, range, skewness, kurtosis, boxplot and the number of outliers of a chosen attribute of this data set
- Bayes classification implementation
- Finding the predicted category of a new sample data that is given by the user

I will present the following topics in this report:

- Implementation of the main code
- Organization of the functions;
- Interpretation of the skewness and kurtosis of attributes;
- User's manual;

The Matlab Code

In my implementation, I created a simple user interface for better functionality for the user –which I will present later in this report. For reading the data set for using in the code, I used **importdata()** function as shown in Figure 2. I took the data matrix using **'data'** feature and attribute names using the **'textdata'** feature that is created after using this function. 'textdata' is important for getting the attribute names to be used later.

```
importedData=importdata('BreastCancerCoimbra.csv');
DATASET=importedData.data;
HEADER=importedData.textdata;
```

(Figure 2)

I created a function called **centeredDataMatrix** for finding the centered data matrix as shown in Figure 3. This function accepts one input which is the data matrix, and it returns one output which is the centered data matrix of this given data matrix by subtracting the mean matrix from it.

```
function out=centeredDataMatrix(matrix)
    onesMatrix=ones(size(matrix,1),1);
    meanMatrix=mean(matrix);
    out=matrix-onesMatrix*meanMatrix;
end
```

(Figure 3)

For finding the number of outliers, I created a function called **outlierCount** which accepts one input that is a vector and returns the number of outliers. The function finds the first and third quartile of the vector, the interquartile, lower limit, upper limit. Then it counts the objects that are greater than the upper limit or less than the lower limit as shown in Figure 4.

```
function out=outlierCount(vector)
    Q25=prctile(vector,25);
    Q75=prctile(vector,75);
    interQ=Q75-Q25;
    LL=Q25-1.5*interQ;
    UL=Q75+1.5*interQ;
    count=0;
    for i = 1:length(vector)
        if (vector(i)>UL) | (vector(i)<LL)
            count=count+1;
        end
    end
    out=count;
end
```

(Figure 4)

For finding the mean, median, sum, max, range, skewness, kurtosis, boxplot and the number of outliers of a specific column (by its index), I let the user choose first a column (shown in Figure 4), then choose which of these to find (shown in Figure 5). If the user enters a value that is greater than the number of attributes or less than 1, the code will give a warning and ask again by returning to beginning of the loop that contains the code. After selecting the column index, the user will choose which information he/she wants to retrieve from the menu.

```
fprintf(['\n1. Age\n2. BMI\n3. Glucose\n4. Insulin\n5. HOMA\n6. Leptin\n7. Adiponectin\n' ...
    '8. Resistin\n9. MCP.1\n10. Classification\n']);
str=sprintf('Please choose one specific column(values between 1-%d): ', size(DATASET,2));
column=input(str);
if (column<1) || (column>size(DATASET,2))
    fprintf('Invalid input for column number!\n')
    continue
end
```

(Figure 5)

```
while true
    fprintf(['\n---Operations---\n0. Return back to beginning page\n' ...
        '1. Find the mean\n2. Find the median\n3. Find the sum\n4. Find the maximum value\n' ...
        '5. Find the range\n6. Find the skewness\n7. Find the kurtosis\n8. Display the boxplot\n' ...
        '9. Find the number of outliers'])
    opt2=input('\nPlease choose one of the above for performing on the specified attribute: ');
    fprintf('\n')
    switch opt2
        case 0
            disp('Exiting from the column operations...')
            break
        case 1
            MEAN=mean(DATASET(:,column));
            fprintf('\nThe mean of the ''%s'' attribute is: %f\n', HEADER{column},MEAN)
        case 2
            MEDIAN=prctile(DATASET(:,column),50);
            fprintf('The median of the ''%s'' attribute is: %f\n', HEADER{column}, MEDIAN)
        case 3
            SUM=sum(DATASET(:,column));
            fprintf('The summation of all object in the ''%s'' attribute is: %f\n', HEADER{column}, SUM)
        case 4
            MAX=max(DATASET(:,column));
            fprintf('The maximum value of the ''%s'' attribute is: %f\n', HEADER{column}, MAX)
        case 5
            RANGE=range(DATASET(:,column));
            fprintf('The range of the ''%s'' attribute is: %f\n', HEADER{column}, RANGE)
        case 6
            SKEW=skewness(DATASET(:,column));
            fprintf('The skewness of the ''%s'' attribute is: %f\n', HEADER{column}, SKEW)
        case 7
            KURTOSIS=kurtosis(DATASET(:,column));
            fprintf('The kurtosis of the ''%s'' attribute is: %f\n', HEADER{column}, KURTOSIS)
        case 8
            fprintf('The boxplot of the ''%s'' attribute is given above...\n', HEADER{column})
            col=DATASET(:,column);
            boxplot(col)
        case 9
            outCount=outlierCount(DATASET(:,column));
            fprintf('The number of outliers of the ''%s'' attribute is: %d\n', HEADER{column}, outCount)
        otherwise
            fprintf('Unvalid option!\nPlease choose a valid option...\n')
    end
```

(Figure 6)

For implementing the Bayes Classifier algorithm, I created three functions called **bayesClassifier**, **bayesClassifierHelper** and **FX**. **FX** function gives the probability density at the new tuple that is entered by the user. This is for using it later in **ŷ** which is the 'Bayes rule' shown in Figure 7. The formula of f(x) is shown in Figure 7 and the implementation of it is shown in Figure 8.

$$\hat{y} = \arg\max_{c_i} \left\{ \hat{f}_i(\mathbf{x}) P(c_i) \right\}$$

where

$$\hat{f}_i(\mathbf{x}) = \hat{f}(\mathbf{x}|\hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\Sigma}}_i) = \frac{1}{(\sqrt{2\pi})^d \sqrt{|\hat{\boldsymbol{\Sigma}}_i|}} \exp\left\{ -\frac{(\mathbf{x} - \hat{\boldsymbol{\mu}}_i)^T \hat{\boldsymbol{\Sigma}}_i^{-1} (\mathbf{x} - \hat{\boldsymbol{\mu}}_i)}{2} \right\}$$

and

$$P(c_i|\mathbf{x}) = \frac{\hat{f}_i(\mathbf{x}) P(c_i)}{\sum_{j=1}^{k} \hat{f}_j(\mathbf{x}) P(c_j)}$$

(Figure 7)

For implementation of $(\mathbf{x} - \hat{\boldsymbol{\mu}}_i)^T \hat{\boldsymbol{\Sigma}}_i^{-1} (\mathbf{x} - \hat{\boldsymbol{\mu}}_i)$part, I modified the formula a little. Since the tuple that is entered by the used will be a row vector, I used the (x-μ) part as a row vector instead of column vector originally in the 'Bayes rule'. That leads to changing the transposes in the implementation.

```
function out=FX(X, MEAN, E, d)
    mat=X-MEAN;
    inv_E=inv(E);
    firstPart=1/((sqrt(2*pi)^d)*sqrt(det(E)));
    seconfPart=-1*(((mat)*(inv_E)*(mat'))/2);
    out=(firstPart)*exp(seconfPart);
end
```

(Figure 8)

The **bayesClassifierHelper** function simply outputs the required parameters (P(ci) is the prior probability, μ is the mean, and **Σ** is the covariance matrix) need to be returned in the 8th step of the Bayes Classifier algorithm (shown in Figure 9).

BAYESCLASSIFIER ($\mathbf{D} = \{(\mathbf{x}_j, y_j)\}_{j=1}^{n}$):
1 **for** $i = 1, \ldots, k$ **do**
2 $\quad \mathbf{D}_i \leftarrow \{\mathbf{x}_j \mid y_j = c_i, j = 1, \ldots, n\}$ // class-specific subsets
3 $\quad n_i \leftarrow |\mathbf{D}_i|$ // cardinality
4 $\quad \hat{P}(c_i) \leftarrow n_i/n$ // prior probability
5 $\quad \hat{\boldsymbol{\mu}}_i \leftarrow \frac{1}{n_i} \sum_{\mathbf{x}_j \in \mathbf{D}_i} \mathbf{x}_j$ // mean
6 $\quad \mathbf{Z}_i \leftarrow \mathbf{D}_i - \mathbf{1}_{n_i} \hat{\boldsymbol{\mu}}_i^T$ // centered data
7 $\quad \hat{\boldsymbol{\Sigma}}_i \leftarrow \frac{1}{n_i} \mathbf{Z}_i^T \mathbf{Z}_i$ // covariance matrix
8 **return** $\hat{P}(c_i), \hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\Sigma}}_i$ *for all* $i = 1, \ldots, k$

(Figure 9)

The implementation is shown below in Figure 10. The first parameter that is denoted by **matrix** is the main data set itself. For the second step of the Bayes Classifier algorithm, we should gather the tuples that have the same label in the classification attribute. The second parameter of this function is the gathered tuples denoted by **class** which is a matrix with 9 columns –without the classification attribute.

```
function [PROB,MEAN,E]=bayesClassifierHelper(matrix,class)
    rowCountClass=size(class,1);
    rowCountDataset=size(matrix,1);
    PROB=rowCountClass/rowCountDataset;
    MEAN=mean(class);
    centered=centeredDataMatrix(class);
    E=(1/rowCountClass)*(centered')*centered;
end
```

(Figure 10)

For the **bayesClassifier** function (shown in Figure 12), I first created the classes. In my implementation, if the label is '1', then the code adds it to the first class which is the class of 'healthy controls'. If it is '2', then the code adds it to the second class which is the class of 'breast cancer patients'. The tuples added to the classes do not have the classification attribute. After creating the class matrices, I followed the other steps of the Bayes Classifier algorithm (shown in Figure 11).

TESTING (**x and** $\hat{P}(c_i)$, $\hat{\boldsymbol{\mu}}_i$, $\hat{\boldsymbol{\Sigma}}_i$, **for all** $i \in [1, k]$):
9 $\hat{y} \leftarrow \arg\max_{c_i} \{f(\mathbf{x}|\hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\Sigma}}_i) \cdot P(c_i)\}$
10 **return** $\hat{y}$

(Figure 11)

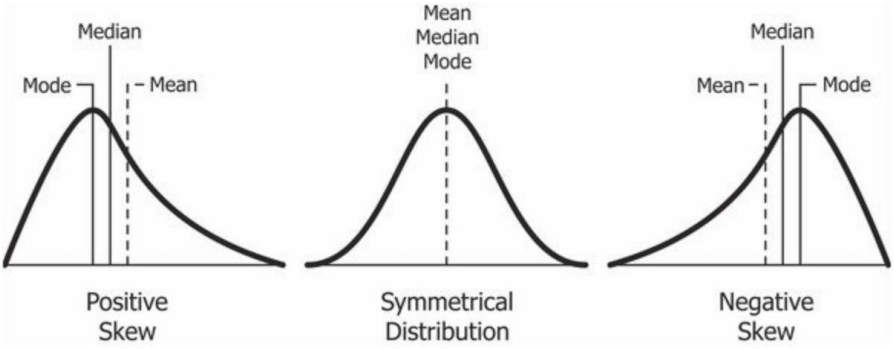
I found the required parameters using **bayesClassifierHelper**, then I found the probabilities of each class then multiplied them with their prior probabilities for finding *P*(ci | x). After finding *P*(ci | x), for each class, I added them to an array for later comparing them. I found the maximum value of this array. Then, I found the index of this value –which is either 1 or 2. Which means indices indicate the value of labels. So, I used the indices directly for assigning the values to the output **CLASSIFICATION**.

```
function CLASSIFICATION=bayesClassifier(matrix, X)
    class1=[];
    class2=[];
    colNumber=size(matrix,2);
    c=colNumber-1;

    for i=1:size(matrix,1)
        if matrix(i,end)==1
            class1(end+1,:)=matrix(i, 1:c);
        elseif matrix(i,end)==2
            class2(end+1,:)=matrix(i, 1:c);
        end
    end

    [PC1, mean1, E1]=bayesClassifierHelper(matrix,class1);
    [PC2, mean2, E2]=bayesClassifierHelper(matrix,class2);

    PROBS=[];

    PROBS(1)=FX(X, mean1, E1, colNumber)*PC1;
    PROBS(2)=FX(X, mean2, E2, colNumber)*PC2;


    CLASSIFICATION = find(PROBS==max(PROBS));


end
```

(Figure 12)

Skewness and Kurtosis

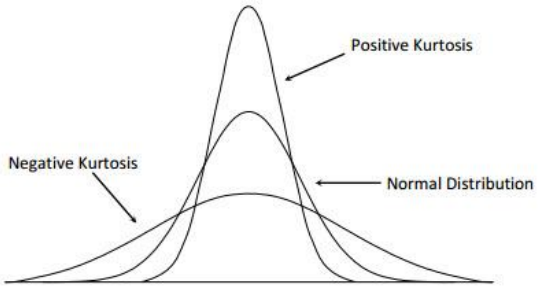
Skewness is a measure of the non-symmetric probability distribution of a random variable. If the skewness of a distribution is less than 0, this means that most values are concentrated on the right of the mean, if it is greater than 0, this means that most values are concentrated on the left of the mean and if it is 0, this means that this distribution is symmetric, most values are concentrated around the mean. Shown in Figure 13.



(Figure 13)

For instance, let us consider the 'Age' attribute in my data set. Its skewness is 0.017601 and its mean is 57.301. Since it is greater than 0, this means that the age of most people is around 57.

Kurtosis is a measure of the 'tailedness' of the probability distribution of a random variable. If the kurtosis of a distribution is less than 3 (negative kurtosis), this means that most values are in a large range around the mean, if it is greater than 3 (positive kurtosis), this means that most values are in a small range around the mean, if it is 3 (normal distribution), this means that most values are in a medium range around the mean. Shown in Figure 14.



(Figure 14)

For instance, let us consider the 'BMI' attribute in my data set. Its kurtosis is 2.065.95 and its mean is 27.582111. Since it is less than 3, this means that the 'BMI' value of most people can vary in a large range around 27.582111.

For interpretation of skewness and kurtosis I let the user choose one attribute or all attributes. For all attributes, I wrote the following code (Figure 15).

```
fprintf('\n-SKEWNESS INTERPRETATION-\n')
for i=1:size(DATASET,2)-1
    fprintf('\nThe skewness value of ''%s'' is: %f\nIndicates that ', HEADER{i}, skewness(DATASET(:,i)))
    if skewness(DATASET(:,i))<0
        fprintf("most of the peoples' '%s' is greater than %f", HEADER{i}, mean(DATASET(:,i)))
    elseif skewness(DATASET(:,i))>0
        fprintf("most of the peoples' '%s' is less than %f", HEADER{i}, mean(DATASET(:,i)))
    else
        fprintf("most of the peoples' '%s' is around %f", HEADER{i}, mean(DATASET(:,i)))
    end
    fprintf('\n')
end

fprintf('\n-KURTOSIS INTERPRETATION-\n')
for i=1:size(DATASET,2)-1
    fprintf('\nThe kurtosis value of ''%s'' is: %f\nIndicates that ', HEADER{i}, kurtosis(DATASET(:,i)))
    if kurtosis(DATASET(:,i))>3
        fprintf("most of the peoples' '%s' is in a small range that is around %f", HEADER{i}, mean(DATASET(:,i)))
    elseif kurtosis(DATASET(:,i))<3
        fprintf("most of the peoples' '%s' is in a large range that is around %f", HEADER{i}, mean(DATASET(:,i)))
    else
        fprintf("most of the peoples' '%s' is around %f", HEADER{i}, mean(DATASET(:,i)))
    end
    fprintf('\n')
end
```

(Figure 15)

For interpretation of skewness and kurtosis of a chosen attribute, I wrote the following code (Figure 16).

```
case 6
    SKEW=skewness(DATASET(:,column));
    fprintf('The skewness of the given attribute is: %f\n', SKEW)
    if SKEW<0
        fprintf("Most of the peoples' %s is greater than %f", HEADER{column}, mean(DATASET(:,column)))
    elseif SKEW>0
        fprintf("Most of the peoples' %s is less than %f", HEADER{column}, mean(DATASET(:,column)))
    else
        fprintf("Most of the peoples' %s is around %f", HEADER{column}, mean(DATASET(:,column)))
    end
case 7
    KURTOSIS=kurtosis(DATASET(:,column));
    fprintf('The kurtosis of the given attribute is: %f\n', KURTOSIS)
    if KURTOSIS>3
        fprintf("most of the peoples' '%s' is in a small range that is around %f", HEADER{column}, mean(DATASET(:,column)))
    elseif KURTOSIS<3
        fprintf("most of the peoples' '%s' is in a large range that is around %f", HEADER{column}, mean(DATASET(:,column)))
    else
        fprintf("most of the peoples' '%s' is around %f", HEADER{column}, mean(DATASET(:,column)))
```

(Figure 16)

User's Manual

I created a simple user's manual as I mentioned before in this report by using switch-case structure. When we execute the Matlab code, the first things that greet us are the introduction of the dataset and after that, a menu where the user can choose her action as shown in Figure 17.

```
>> BreastCancerCoimbra

Data Analysis 2022-2023 Fall Project - Marya Aktas 200444001

The dataset that is used in this code is 'Breast Cancer Coimbra'
In this dataset, there are 10 predictors, all quantitative, and a binary dependent variable, indicating the presence or absence of breast cancer.
The predictors are anthropometric data and parameters which can be gathered in routine blood analysis.
In classification attribute, 1 indicates 'Healthy controls' and 2 indicates 'Patients'


---Menu---

0. Exit the code
1. Find the centered data matrix
2. Retrieve some informations on a specific column
3. Append a new sample data and finding its class
4. Display the dataset
5. Interpretation of the skewness and the kurtosis of all attributes

Please choose one of the above:
```

(Figure 17)

The user should choose an integer in the [0-5] range. Option 0 exits the code -by breaking the loop. Option 1 displays the centered data matrix of the data set shown in Figure 18.

```
Please choose one of the above: 1

The centered data matrix of the given dataset is:
   1.0e+03 *

   -0.0093   -0.0041   -0.0278   -0.0073   -0.0022   -0.0178   -0.0005   -0.0067   -0.1175   -0.0006
    0.0257   -0.0069   -0.0058   -0.0069   -0.0020   -0.0178   -0.0048   -0.0107   -0.0659   -0.0006
    0.0247   -0.0045   -0.0068   -0.0055   -0.0017   -0.0087    0.0123   -0.0054    0.0201   -0.0006
    0.0107   -0.0062   -0.0208   -0.0068   -0.0021   -0.0167   -0.0030   -0.0020    0.3936   -0.0006
    0.0287   -0.0065   -0.0058   -0.0065   -0.0019   -0.0199   -0.0054   -0.0041    0.2393   -0.0006
   -0.0083   -0.0047   -0.0058   -0.0068   -0.0020   -0.0198    0.0035   -0.0044   -0.0042   -0.0006
    0.0317   -0.0049   -0.0208   -0.0053   -0.0018   -0.0197   -0.0046   -0.0018    0.7214   -0.0006
    0.0187   -0.0038    0.0202   -0.0035   -0.0008   -0.0223    0.0031   -0.0096   -0.2540   -0.0006
    0.0157   -0.0056   -0.0008   -0.0067   -0.0019   -0.0221    0.0002   -0.0084   -0.3978   -0.0006
```

(Figure 18)

Option 2 asks for an attribute and directs us to another menu which the user can retrieve information about this attribute as shown in Figure 19.

```
Please choose one of the above: 2

1. Age
2. BMI
3. Glucose
4. Insulin
5. HOMA
6. Leptin
7. Adiponectin
8. Resistin
9. MCP.1
10. Classification
Please choose one specific column(values between 1-10):
```

(Figure 19)

After choosing the attribute the ‘Operations’ menu pops up as shown in Figure 20. It asks for which operation the user wants to perform.

```
Please choose one specific column(values between 1-10): 2


---Operations---
0. Return back to beginning page
1. Find the mean
2. Find the median
3. Find the sum
4. Find the maximum value
5. Find the range
6. Find the skewness
7. Find the kurtosis
8. Display the boxplot
9. Find the number of outliers
Please choose one of the above for performing on the specified attribute:
```

(Figure 20)

Some examples of outputs after choosing one of these are shown below:

```
Please choose one of the above for performing on the specified attribute: 2

The median of the 'Glucose' attribute is: 92.000000
```

(Example 1)

```
Please choose one of the above for performing on the specified attribute: 5

The range of the 'Insulin' attribute is: 56.028000
```

(Example 2)

```
Please choose one of the above for performing on the specified attribute: 9

The number of outliers of the 'Glucose' attribute is: 12
```

(Example 3)

If the user chooses skewness or kurtosis, the code also gives the interpretation of them as shown in Figure 21 and 22.

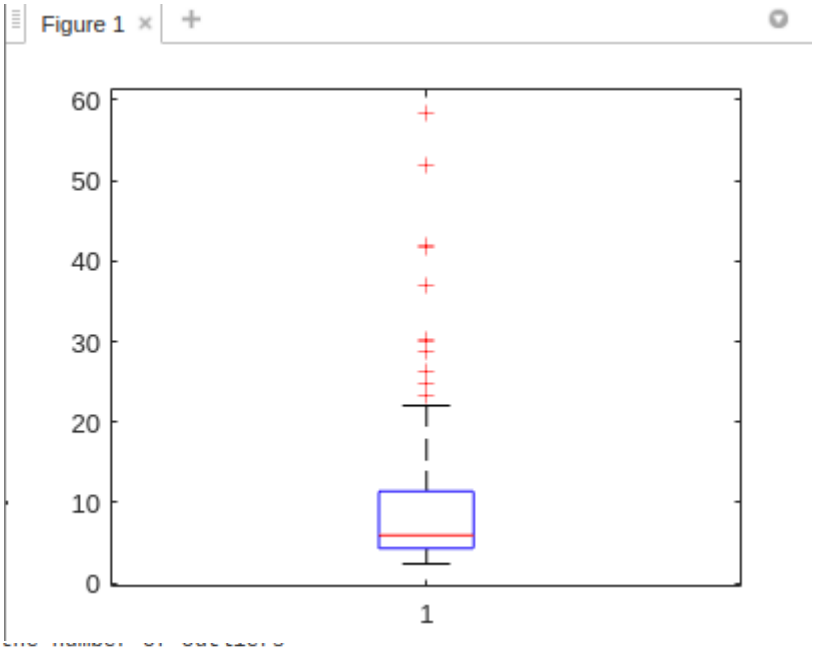
```
Please choose one of the above for performing on the specified attribute: 6

The skewness of the given attribute is: 2.544116
Most of the peoples' Insulin is less than 10.012086
```

(Figure 21)

```
Please choose one of the above for performing on the specified attribute: 7

The kurtosis of the given attribute is: 10.131281
most of the peoples' 'Insulin' is in a small range that is around 10.012086
```

(Figure 22)

We can also know the accuracy of finding the outliers of my code by looking at the boxplot of the selected attribute. The boxplot of 'Insulin' attribute is shown in Figure 23 and the number of outliers of it is shown in Figure 24. We can count the outliers from the boxplot by looking at the '+' symbols.




```
Please choose one of the above for performing on the specified attribute:
8

The boxplot of the 'Insulin' attribute is given above...
```

(Figure 23)

```
Please choose one of the above for performing on the specified attribute:
9

The number of outliers of the 'Insulin' attribute is: 11
```

(Figure 24)

After returning to the beginning page by choosing option 0, the user may add a new tuple to the data set without knowing its category by choosing option 3. For testing my **bayesClassifier** function, I entered new tuples that are the same as some tuples in original data set to see if the classes of them are true. I used a tuple that has '1' label in its classification attribute in Figure 25. The program predicts its category successfully.

```
Please choose one of the above: 3
Please enter the values of the predictors of the new person (as a data vector with size 1x9) to see which class it belongs to: [44 20.76 86 7.553 1.6 14.09 20.32 7.64 63.61]
The predicted category of this person is: 1
... meaning that this person is not a breast cancer patient.

The new sample data added to the dataset successfully...
```

(Figure 25)

I also let the user display the data matrix whenever she wants in option 4 as shown in Figure 26.

```
Please choose one of the above: 4
This data matrix consist of numeric values with size 117x10
The numeric values of displayed dataset is divided by 1.0e+03

   1.0e+03 *

    0.0480    0.0235    0.0700    0.0027    0.0005    0.0088    0.0097    0.0080    0.4171    0.0010
    0.0830    0.0207    0.0920    0.0031    0.0007    0.0088    0.0054    0.0041    0.4688    0.0010
    0.0820    0.0231    0.0910    0.0045    0.0010    0.0179    0.0224    0.0093    0.5547    0.0010
    0.0680    0.0214    0.0770    0.0032    0.0006    0.0099    0.0072    0.0128    0.9282    0.0010
    0.0860    0.0211    0.0920    0.0035    0.0008    0.0067    0.0048    0.0106    0.7739    0.0010
    0.0490    0.0229    0.0920    0.0032    0.0007    0.0068    0.0137    0.0103    0.5304    0.0010
```

(Figure 26)

Option 5 gives the interpretation of skewness and kurtosis of all attributes as shown in Figure 27 and Figure 28.

```
-KURTOSIS INTERPRETATION-

The kurtosis value of 'Age' is: 2.005982
Indicates that most of the peoples' 'Age' is in a large range that is around 57.188034

The kurtosis value of 'BMI' is: 2.054826
Indicates that most of the peoples' 'BMI' is in a large range that is around 27.523802

The kurtosis value of 'Glucose' is: 11.645508
Indicates that most of the peoples' 'Glucose' is in a small range that is around 97.692308

The kurtosis value of 'Insulin' is: 10.229672
Indicates that most of the peoples' 'Insulin' is in a small range that is around 9.991068

The kurtosis value of 'HOMA' is: 19.969129
```

(Figure 27)

```
-SKEWNESS INTERPRETATION-

The skewness value of 'Age' is: 0.033919
Indicates that most of the peoples' 'Age' is less than 57.188034

The skewness value of 'BMI' is: 0.178092
Indicates that most of the peoples' 'BMI' is less than 27.523802

The skewness value of 'Glucose' is: 2.573844
Indicates that most of the peoples' 'Glucose' is less than 97.692308
```

(Figure 28)

Conclusion

While implementing this program with Matlab, I improved my skills in algorithm development and matrix management. I reinforced my function communication skills and how I can interpret information about a data set. I also performed mathematical and statistical operations on a real-valued data set. I could see how I can apply theorical knowledge of data analysis course.