Course: Probability, Statistics and Machine Learning– ENDG 319
Deliverable #: 2
Student Name: Gerard Ledoux Mendjemo Ngangom
Lecture Section: L01
Date submitted: Oct 15, 2023

ENDG 319 - Fall 23

CURE Project – Deliverable 2

>>

Graphical data analysis to develop a machine-learning model

Instructions:

- This deliverable is worth 3% of your overall grade and should be submitted individually.
- The assignment will require you to use Python **and you will have to paste screenshots of your work in a Word file to show your work that involves coding**. Please keep the original Word file. Once done, convert the Word file into pdf with file name: Last Name_First Name_UCID_CURE Deliverable 2.
- Submit the pdf file to Assessments > Dropbox > 'CURE Project Deliverable 2' by Oct 15, 2023, 11:59 pm (MT).
- You must submit this deliverable on time to be able to submit the upcoming deliverables.

>>

Research skill:

(i) Dealing with real-world data

(ii) Graphical analysis of data, interpreting figure

(iii) Critical thinking - drawing conclusions from data

(iv) Use of engineering tools – python

(v) Developing a research question

Relevant course content: Descriptive Statistics (Ch 1), Machine learning – classification

>>

Exploring the breast_cancer dataset in sklearn

In the breast_cancer database there are 30 features and 2 classes, as shown below.

```
In [1]: import sklearn.datasets
        cancer = sklearn.datasets.load_breast_cancer()

In [2]: cancer.keys()

Out[2]: dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names', 'filename'])

In [3]: print(cancer.feature_names)

['mean radius' 'mean texture' 'mean perimeter' 'mean area'
 'mean smoothness' 'mean compactness' 'mean concavity'
 'mean concave points' 'mean symmetry' 'mean fractal dimension'
 'radius error' 'texture error' 'perimeter error' 'area error'
 'smoothness error' 'compactness error' 'concavity error'
 'concave points error' 'symmetry error' 'fractal dimension error'
 'worst radius' 'worst texture' 'worst perimeter' 'worst area'
 'worst smoothness' 'worst compactness' 'worst concavity'
 'worst concave points' 'worst symmetry' 'worst fractal dimension']

In [4]: print(cancer.target_names)

['malignant' 'benign']
```

More information is available in the description. Read the following snippet from the description.

```
In [5]: print(cancer.DESCR)

    :Class Distribution: 212 - Malignant, 357 - Benign

    :Creator:  Dr. William H. Wolberg, W. Nick Street, Olvi L. Mangasarian

    :Donor: Nick Street

    :Date: November, 1995

This is a copy of UCI ML Breast Cancer Wisconsin (Diagnostic) datasets.
https://goo.gl/U2Uwz2

Features are computed from a digitized image of a fine needle
aspirate (FNA) of a breast mass.  They describe
characteristics of the cell nuclei present in the image.
```

Task 1

Create a dataframe variable 'a' with this dataset. This dataframe should have all the 569 instances, 30 features and the class of 569 instances as 0 (Malignant) or 1 (Benign). The column that contains the classes should be labeled as 'typeofcancer'. Show the output of the following input:

```
In [13]:  a.shape
```

[Hints: the outputs should be same as below.

```
Out[13]: (569, 31)
```

(b) Now create a dataframe variable 'df' by slicing dataframe 'a'. The new dataframeme 'df' should have all the instances, their labels but with the following three features: mean radius, mean perimeter and mean area. [Hints: use .iloc method to extract necessary columns from 'a']

(i) Show the first two rows.

[Hint: The output should be same as below.

Out[16]:

	mean radius	mean perimeter	mean area	typeofcancer
0	17.99	122.8	1001.0	0
1	20.57	132.9	1326.0	0

(ii) Show the rows with indexes 17, 18, 19, 20, 21.

Task 2

Suppose we want to explore the possibility of developing a machine learning model that can diagnose a new patient's cancer condition as benign or malignant from the features in df.

(i) As a first step, you want to do some graphical analysis. Write the code to generate the following figure (Figure 1). Show screenshot of the code (input) and the figure (output) from your work. You are free to choose your favorite data marker and color in your figure.

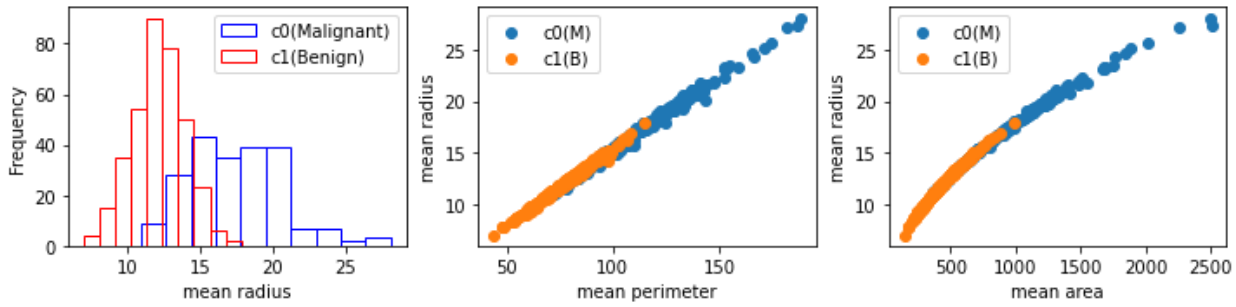
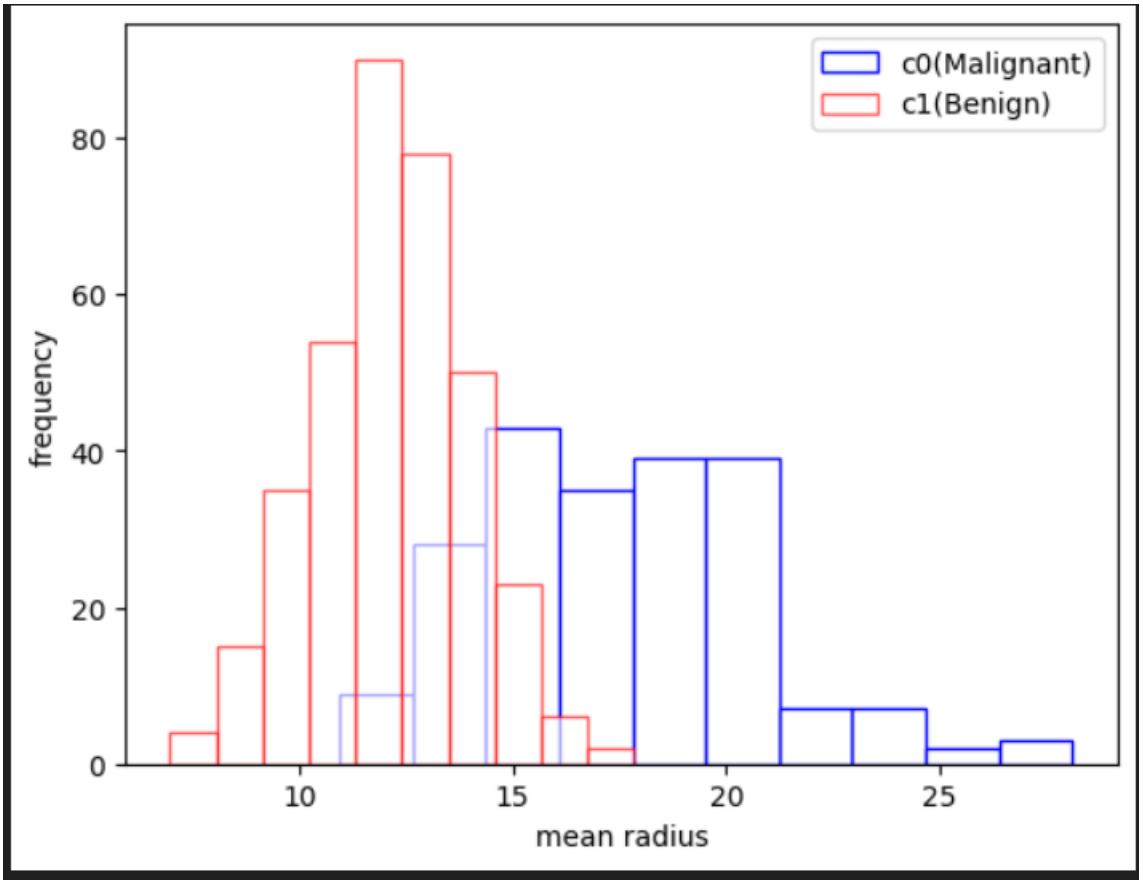


Figure 1: From left of right: histogram of 'mean radius' data for each class, scatter plot of 'mean radius' versus 'mean perimeter', scatter plot of 'mean radius' versus 'mean area'.

Code graph1:

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()

import numpy as np
mean_radius = np.array(df.iloc[:, 3])

pos = 0
mean_radius_malignant = np.array([])
mean_radius_benign = np.array([])

for row in mean_radius:
    if row == 0:
        mean_radius_malignant = np.append(mean_radius_malignant, df.iloc[pos, 0])
    else:
        mean_radius_benign = np.append(mean_radius_benign, df.iloc[pos, 0])
    pos += 1

plt.hist(mean_radius_malignant, bins=10, color='w', edgecolor='b', alpha=1.0, label='c0(Malignant)')
plt.hist(mean_radius_benign, bins=10, color = 'w', edgecolor='r', alpha=0.65, label='c1(Benign)')


plt.xlabel("mean radius")
plt.ylabel("frequency")
plt.legend()
plt.show()
```

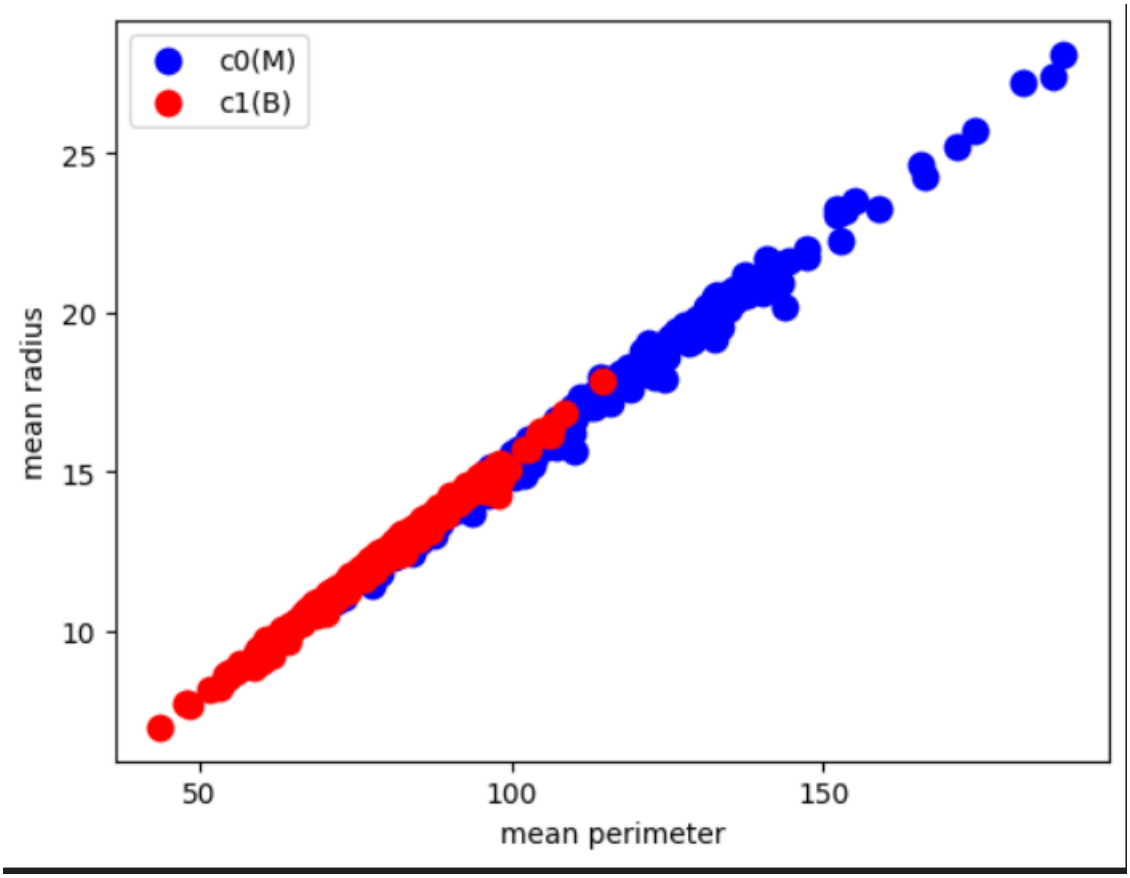
Graph1:



Code graph2:

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()

import numpy as np
mean_radius = np.array(df.iloc[:, 3])
pos = 0
mean_radius_malignant = np.array([])
mean_radius_benign = np.array([])
mean_perimeter_malignant = np.array([])
mean_perimeter_benign = np.array([])

for row in mean_radius:
    if row == 0:
        mean_radius_malignant = np.append(mean_radius_malignant, df.iloc[pos, 0])
        mean_perimeter_malignant = np.append(mean_perimeter_malignant, df.iloc[pos, 1])
    else:
        mean_radius_benign = np.append(mean_radius_benign, df.iloc[pos, 0])
        mean_perimeter_benign = np.append(mean_perimeter_benign, df.iloc[pos, 1])
    pos += 1

ax.scatter(mean_perimeter_malignant, mean_radius_malignant, c='b', marker='o', s=80, label="c0(M)")
ax.scatter(mean_perimeter_benign, mean_radius_benign, c='r', marker='o', s=80, label="c1(B)")


plt.xticks([50, 100, 150],['50', '100', '150'])
plt.xlabel("mean perimeter")
plt.ylabel("mean radius")
plt.legend()
plt.show()
```

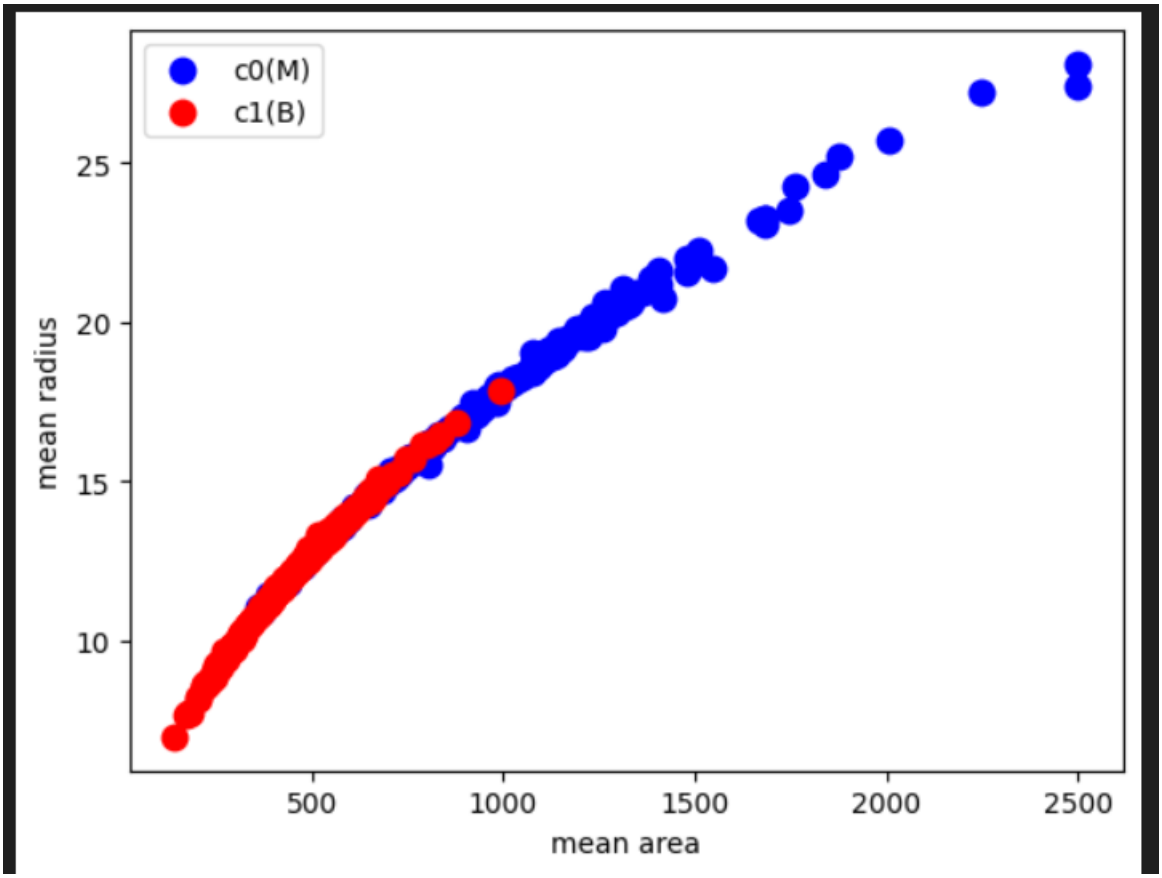
Graph2:



Code graph3:

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()

import numpy as np
mean_radius = np.array(df.iloc[:, 3])
pos = 0
mean_radius_malignant = np.array([])
mean_radius_benign = np.array([])
mean_area_malignant = np.array([])
mean_area_benign = np.array([])

for row in mean_radius:
    if row == 0:
        mean_radius_malignant = np.append(mean_radius_malignant, df.iloc[pos, 0])
        mean_area_malignant = np.append(mean_area_malignant, df.iloc[pos, 2])
    else:
        mean_radius_benign = np.append(mean_radius_benign, df.iloc[pos, 0])
        mean_area_benign = np.append(mean_area_benign, df.iloc[pos, 2])
    pos += 1

ax.scatter(mean_area_malignant, mean_radius_malignant, c='b', marker='o', s=80, label="c0(M)")
ax.scatter(mean_area_benign, mean_radius_benign, c='r', marker='o', s=80, label="c1(B)")


#plt.xticks([50, 100, 150],['50', '100', '150'])
plt.xlabel("mean area")
plt.ylabel("mean radius")
plt.legend()
plt.show()
```

Graph3:



(ii) Briefly describe what each of the subplots in Figure 1 reveal about the data.

Answer

The histogram illustrates that the mean radius of Malignant tumors exhibits a wider range, making it more sparsely distributed in comparison to Benign tumors.

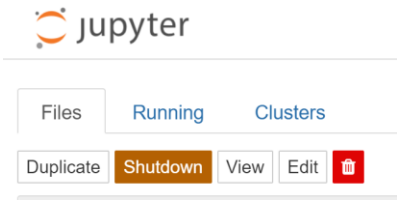
In the second scatter plot, it becomes apparent that the majority of mean perimeter and mean radius values for Malignant tumors are larger than those for Benign tumors. Additionally, a smooth, straight line can be drawn, suggesting a linear relationship between mean radius and mean perimeter for both Malignant and Benign cases.

The third scatter plot highlights that most of the mean area and mean radius values for Malignant tumors are greater than those for Benign tumors. Towards the end of the plot, there is a gentle curve, indicating a logarithmic relationship between the values of Malignant and Benign cases for mean area and mean radius..

Task 3

Save the file generated in Task 2. Copy it and modify the code to generate the following figure (Figure 2).

Note: In juputer notebook you can use the 'Duplicate' button to create a copy of a file as shown below. The 'Duplicate' button will be available when you select the file you want to copy.



(i) Generate the following figure (Figure 2). Show screenshot(s) of the code (input) and the figure (output) from your work. You are free to choose your favorite data marker and color in your figure.

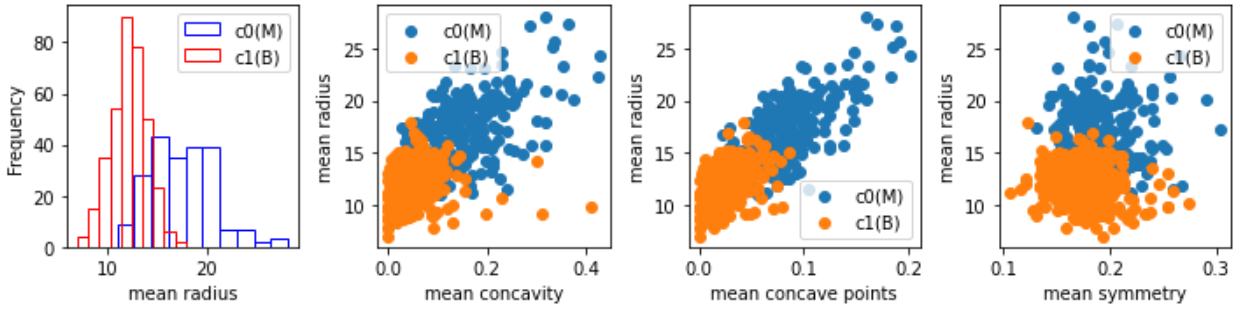
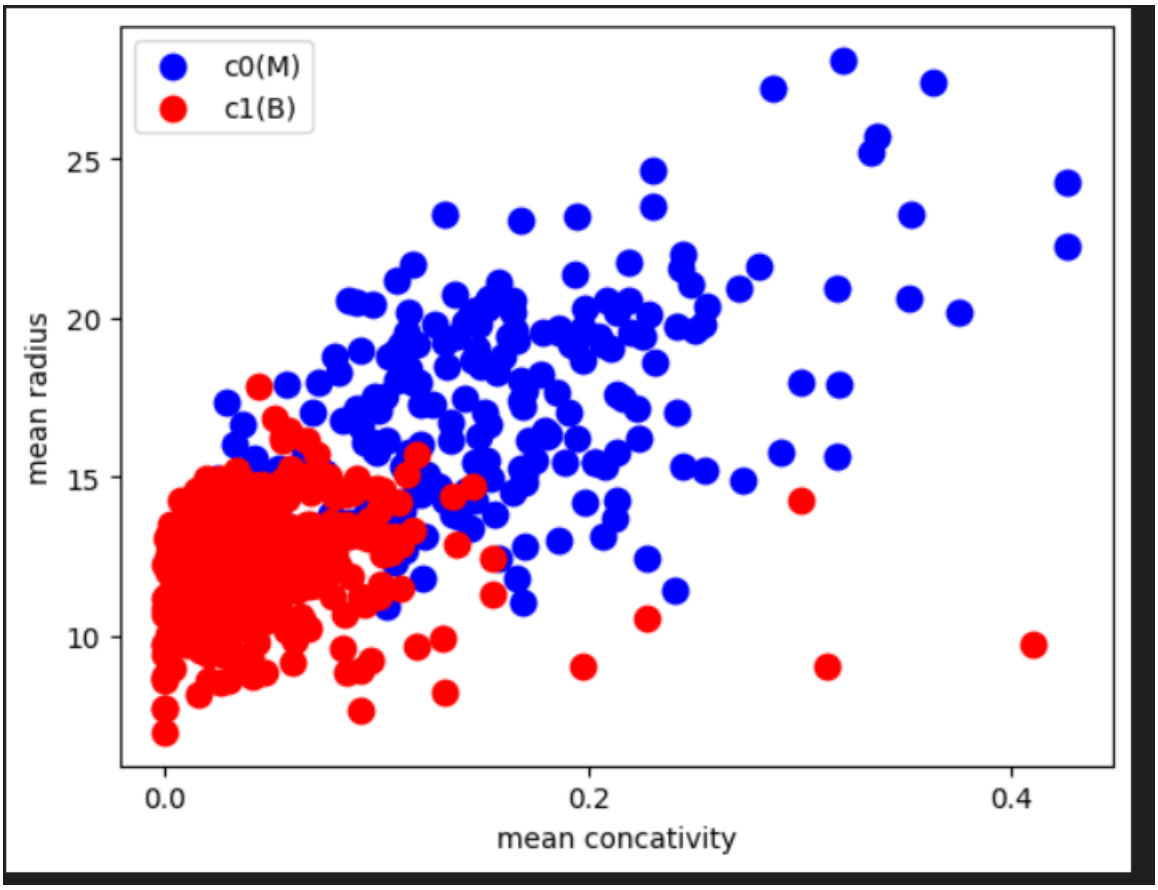


Figure 2: From left of right: histogram of 'mean radius' data for each class, scatter plot of 'mean radius' versus 'mean concavity', scatter plot of 'mean radius' versus 'mean concave points', and scatter plot of 'mean radius' versus 'mean symmetry'.

Code graph1:

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()

import numpy as np
mean_radius = np.array(df.iloc[:, 4])
pos = 0
mean_radius_malignant = np.array([])
mean_radius_benign = np.array([])
mean_concativity_malignant = np.array([])
mean_concativity_benign = np.array([])

for row in mean_radius:
    if row == 0:
        mean_radius_malignant = np.append(mean_radius_malignant, df.iloc[pos, 0])
        mean_concativity_malignant = np.append(mean_concativity_malignant, df.iloc[pos, 1])
    else:
        mean_radius_benign = np.append(mean_radius_benign, df.iloc[pos, 0])
        mean_concativity_benign = np.append(mean_concativity_benign, df.iloc[pos, 1])
    pos += 1

ax.scatter(mean_concativity_malignant, mean_radius_malignant, c='b', marker='o', s=80, label="c0(M)")
ax.scatter(mean_concativity_benign, mean_radius_benign, c='r', marker='o', s=80, label="c1(B)")


plt.xticks([0.0, 0.2, 0.4],['0.0', '0.2', '0.4'])
plt.xlabel("mean concativity")
plt.ylabel("mean radius")
plt.legend()
plt.show()
```

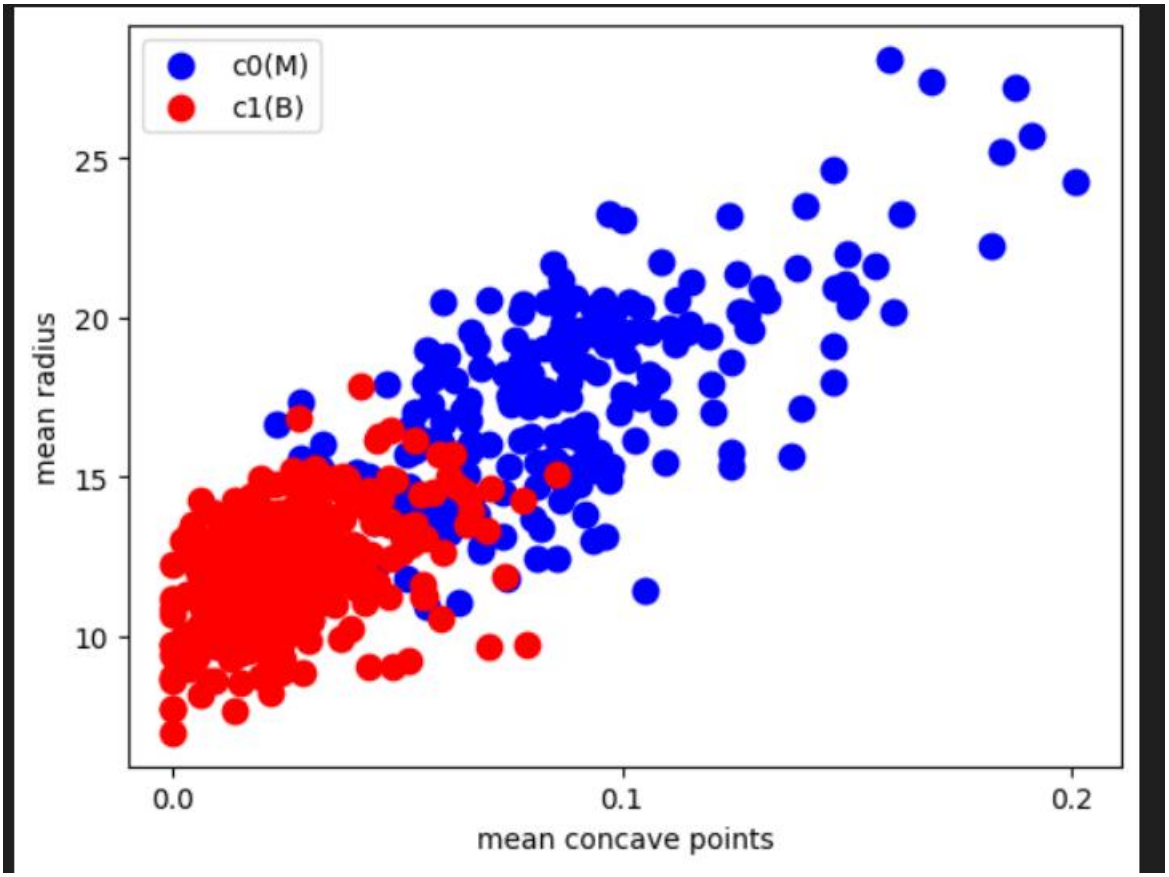
Graph1:



Code graph2:

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()

import numpy as np
mean_radius = np.array(df.iloc[:, 4])
pos = 0
mean_radius_malignant = np.array([])
mean_radius_benign = np.array([])
mean_concave_malignant = np.array([])
mean_concave_benign = np.array([])

for row in mean_radius:
    if row == 0:
        mean_radius_malignant = np.append(mean_radius_malignant, df.iloc[pos, 0])
        mean_concave_malignant = np.append(mean_concave_malignant, df.iloc[pos, 2])
    else:
        mean_radius_benign = np.append(mean_radius_benign, df.iloc[pos, 0])
        mean_concave_benign = np.append(mean_concave_benign, df.iloc[pos, 2])
    pos += 1

ax.scatter(mean_concave_malignant, mean_radius_malignant, c='b', marker='o', s=80, label="c0(M)")
ax.scatter(mean_concave_benign, mean_radius_benign, c='r', marker='o', s=80, label="c1(B)")


plt.xticks([0.0, 0.1, 0.2],['0.0', '0.1', '0.2'])
plt.xlabel("mean concave points")
plt.ylabel("mean radius")
plt.legend()
plt.show()
```

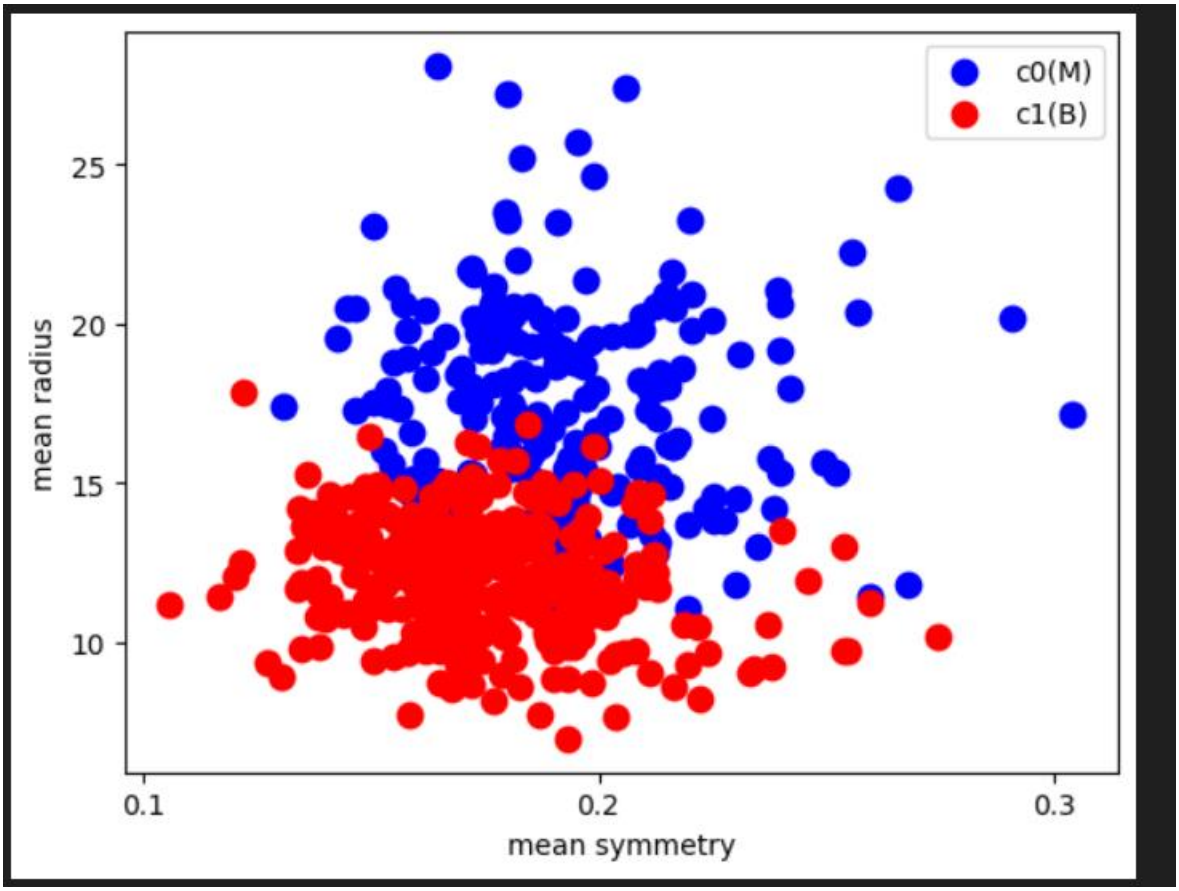
Graph2:



Code graph3:

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()

import numpy as np
mean_radius = np.array(df.iloc[:, 4])
pos = 0
mean_radius_malignant = np.array([])
mean_radius_benign = np.array([])
mean_symmetry_malignant = np.array([])
mean_symmetry_benign = np.array([])

for row in mean_radius:
    if row == 0:
        mean_radius_malignant = np.append(mean_radius_malignant, df.iloc[pos, 0])
        mean_symmetry_malignant = np.append(mean_symmetry_malignant, df.iloc[pos, 3])
    else:
        mean_radius_benign = np.append(mean_radius_benign, df.iloc[pos, 0])
        mean_symmetry_benign = np.append(mean_symmetry_benign, df.iloc[pos, 3])
    pos += 1

ax.scatter(mean_symmetry_malignant, mean_radius_malignant, c='b', marker='o', s=80, label="c0(M)")
ax.scatter(mean_symmetry_benign, mean_radius_benign, c='r', marker='o', s=80, label="c1(B)")


plt.xticks([0.1, 0.2, 0.3],['0.1', '0.2', '0.3'])
plt.xlabel("mean symmetry")
plt.ylabel("mean radius")
plt.legend()
plt.show()
```

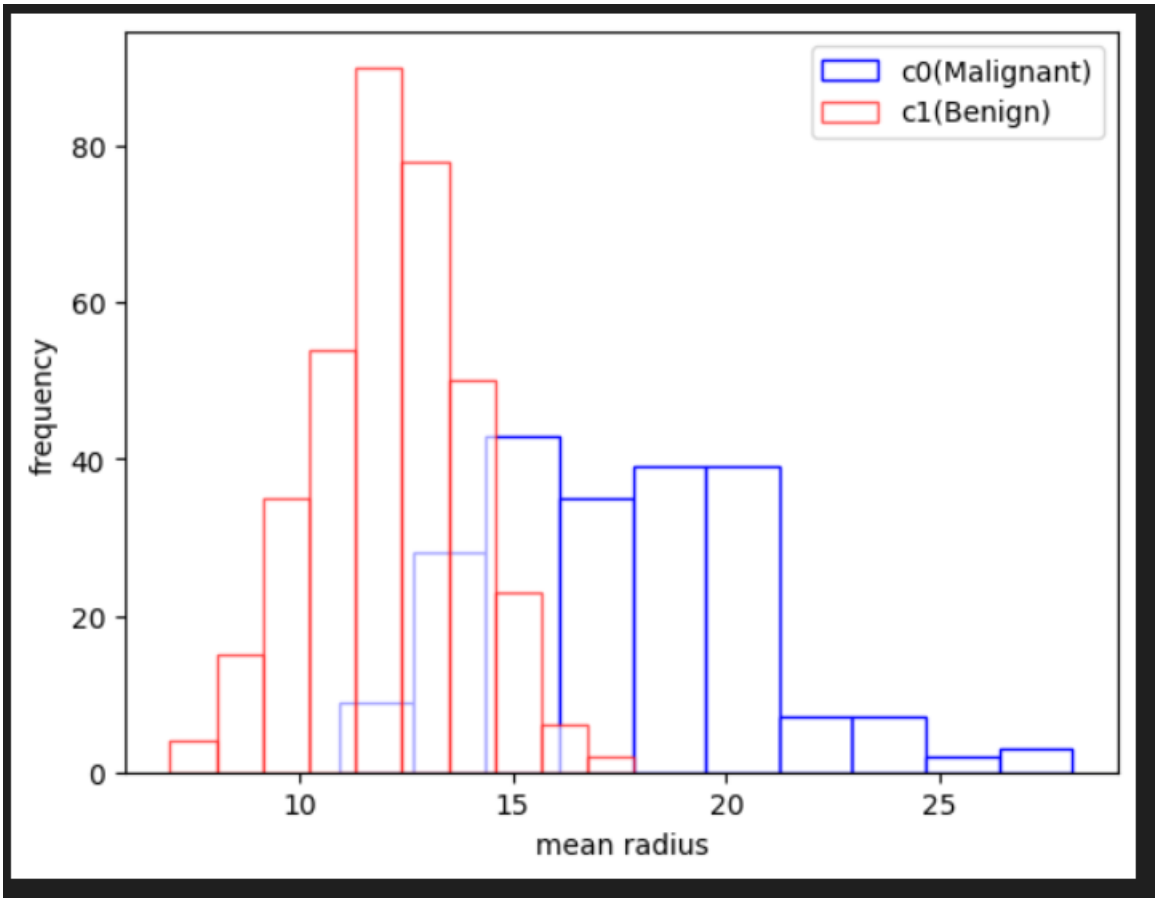
Graph3:



Code graph 4:

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()

import numpy as np
mean_radius = np.array(df.iloc[:, 4])

pos = 0
mean_radius_malignant = np.array([])
mean_radius_benign = np.array([])

for row in mean_radius:
    if row == 0:
        mean_radius_malignant = np.append(mean_radius_malignant, df.iloc[pos, 0])
    else:
        mean_radius_benign = np.append(mean_radius_benign, df.iloc[pos, 0])
    pos += 1

plt.hist(mean_radius_malignant, bins=10, color='w', edgecolor='b', alpha=1.0, label='c0(Malignant)')
plt.hist(mean_radius_benign, bins=10, color = 'w', edgecolor='r', alpha=0.65, label='c1(Benign)')


plt.xlabel("mean radius")
plt.ylabel("frequency")
plt.legend()
plt.show()
```

Graph4:



(ii) Briefly describe what each of the subplots (except the histogram) in Figure 2 reveal about the data.

Answer:

The initial scatter plot demonstrates that a significant portion of benign samples is clustered within a particular range of mean concavity and mean radius, whereas the samples for Malignant cases exhibit greater dispersion.

In the second scatter plot, similar to the first one, it is observed that Malignant samples are more widely dispersed than benign samples. Specifically, around a mean radius of 15 and a mean concave point of 0.1, a clear convergence point can be observed, suggesting the transition from Benign to Malignant samples.

Task 4

Think of a classification problem from your own experience and interest (e.g., distinguishing between two types of things/living beings/weather/personality/vehicle/industry/sports/houses/professionals etc.). Collect some real-

world data to solve that classification problem. Have at least two classes, two attributes and at least 5 instances per class. Create a dataframe with this data and paste a screenshot of it. Also give a short description of the data (number of instances, number of attributes and the list of attributes, list of classes, and creator: Your name). The attributes should be numerical.

The classification problem should be a new one, but the dataset may contain data that are new (generated by you by any rough measurements) or collected from public domain. Please cite the references/sources from which you collect the data. Describe the Data:

Description of data:

- Number of Instances: 10 (5 per class)
- Number of Attributes: 2 (Number of Legs, Body Weight)
- List of Attributes: Number of Legs, Body Weight
- List of Classes: Mammals, Birds
- Creator: Mendjemo Ngangom Gerard Ledoux

```
import pandas as pd

# Create a dictionary with your data
data = {
    'Number of Legs': [4, 4, 4, 4, 4, 2, 2, 2, 2, 2],
    'Body Weight (kg)': [50, 20, 30, 60, 80, 1, 2, 0.5, 0.8, 1.5],
    'Class': ['Mammals', 'Mammals', 'Mammals', 'Mammals', 'Mammals', 'Birds', 'Birds', 'Birds', 'Birds', 'Birds']
}

# Create a DataFrame
df = pd.DataFrame(data)

# Print the DataFrame
print(df)
```

[1] ✓ 0.5s

```
   Number of Legs  Body Weight (kg)    Class
0               4              50.0  Mammals
1               4              20.0  Mammals
2               4              30.0  Mammals
3               4              60.0  Mammals
4               4              80.0  Mammals
5               2               1.0    Birds
6               2               2.0    Birds
7               2               0.5    Birds
8               2               0.8    Birds
9               2               1.5    Birds
```