deliverable2

October 16, 2023

TASK 1

[1]:
```python
import sklearn.datasets
from sklearn.datasets import load_breast_cancer

cancer = load_breast_cancer()
print(cancer.keys())
print(cancer.DESCR)
```

```
dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names',
'filename', 'data_module'])
.. _breast_cancer_dataset:

Breast cancer wisconsin (diagnostic) dataset
--------------------------------------------

**Data Set Characteristics:**

    :Number of Instances: 569

    :Number of Attributes: 30 numeric, predictive attributes and the class

    :Attribute Information:
        - radius (mean of distances from center to points on the perimeter)
        - texture (standard deviation of gray-scale values)
        - perimeter
        - area
        - smoothness (local variation in radius lengths)
        - compactness (perimeter^2 / area - 1.0)
        - concavity (severity of concave portions of the contour)
        - concave points (number of concave portions of the contour)
        - symmetry
        - fractal dimension ("coastline approximation" - 1)

        The mean, standard error, and "worst" or largest (mean of the three
        worst/largest values) of these features were computed for each image,
        resulting in 30 features.  For instance, field 0 is Mean Radius, field
        10 is Radius SE, field 20 is Worst Radius.
    - class:
            - WDBC-Malignant
            - WDBC-Benign

:Summary Statistics:

===================================== ====== ======
                                       Min    Max
===================================== ====== ======
radius (mean):                        6.981  28.11
texture (mean):                       9.71   39.28
perimeter (mean):                     43.79  188.5
area (mean):                          143.5  2501.0
smoothness (mean):                    0.053  0.163
compactness (mean):                   0.019  0.345
concavity (mean):                     0.0    0.427
concave points (mean):                0.0    0.201
symmetry (mean):                      0.106  0.304
fractal dimension (mean):             0.05   0.097
radius (standard error):              0.112  2.873
texture (standard error):             0.36   4.885
perimeter (standard error):           0.757  21.98
area (standard error):                6.802  542.2
smoothness (standard error):          0.002  0.031
compactness (standard error):         0.002  0.135
concavity (standard error):           0.0    0.396
concave points (standard error):      0.0    0.053
symmetry (standard error):            0.008  0.079
fractal dimension (standard error):   0.001  0.03
radius (worst):                       7.93   36.04
texture (worst):                      12.02  49.54
perimeter (worst):                    50.41  251.2
area (worst):                         185.2  4254.0
smoothness (worst):                   0.071  0.223
compactness (worst):                  0.027  1.058
concavity (worst):                    0.0    1.252
concave points (worst):               0.0    0.291
symmetry (worst):                     0.156  0.664
fractal dimension (worst):            0.055  0.208
===================================== ====== ======

:Missing Attribute Values: None

: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.

Separating plane described above was obtained using
Multisurface Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree
Construction Via Linear Programming." Proceedings of the 4th
Midwest Artificial Intelligence and Cognitive Science Society,
pp. 97-101, 1992], a classification method which uses linear
programming to construct a decision tree. Relevant features
were selected using an exhaustive search in the space of 1-4
features and 1-3 separating planes.

The actual linear program used to obtain the separating plane
in the 3-dimensional space is that described in:
[K. P. Bennett and O. L. Mangasarian: "Robust Linear
Programming Discrimination of Two Linearly Inseparable Sets",
Optimization Methods and Software 1, 1992, 23-34].

This database is also available through the UW CS ftp server:

```
ftp ftp.cs.wisc.edu
cd math-prog/cpo-dataset/machine-learn/WDBC/
```

|details-start|
References
|details-split|

- W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extraction
  for breast tumor diagnosis. IS&T/SPIE 1993 International Symposium on
  Electronic Imaging: Science and Technology, volume 1905, pages 861-870,
  San Jose, CA, 1993.
- O.L. Mangasarian, W.N. Street and W.H. Wolberg. Breast cancer diagnosis and
  prognosis via linear programming. Operations Research, 43(4), pages 570-577,
  July-August 1995.
- W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machine learning techniques
  to diagnose breast cancer from fine-needle aspirates. Cancer Letters 77 (1994)
  163-171.

|details-end|

[2]:
```
cancer = load_breast_cancer(return_X_y = True, as_frame = True)

a = cancer[0]
a['typeofcancer'] = cancer[1]
```

[3]:
```
a.shape
```

[3]:
```
(569, 31)
```

[4]:
```
df = a.iloc[:, [0, 2, 3, 30]]
df.iloc[0:2, :] #Show the first two rows
```

[4]:
```
   mean radius  mean perimeter  mean area  typeofcancer
0        17.99           122.8     1001.0             0
1        20.57           132.9     1326.0             0
```

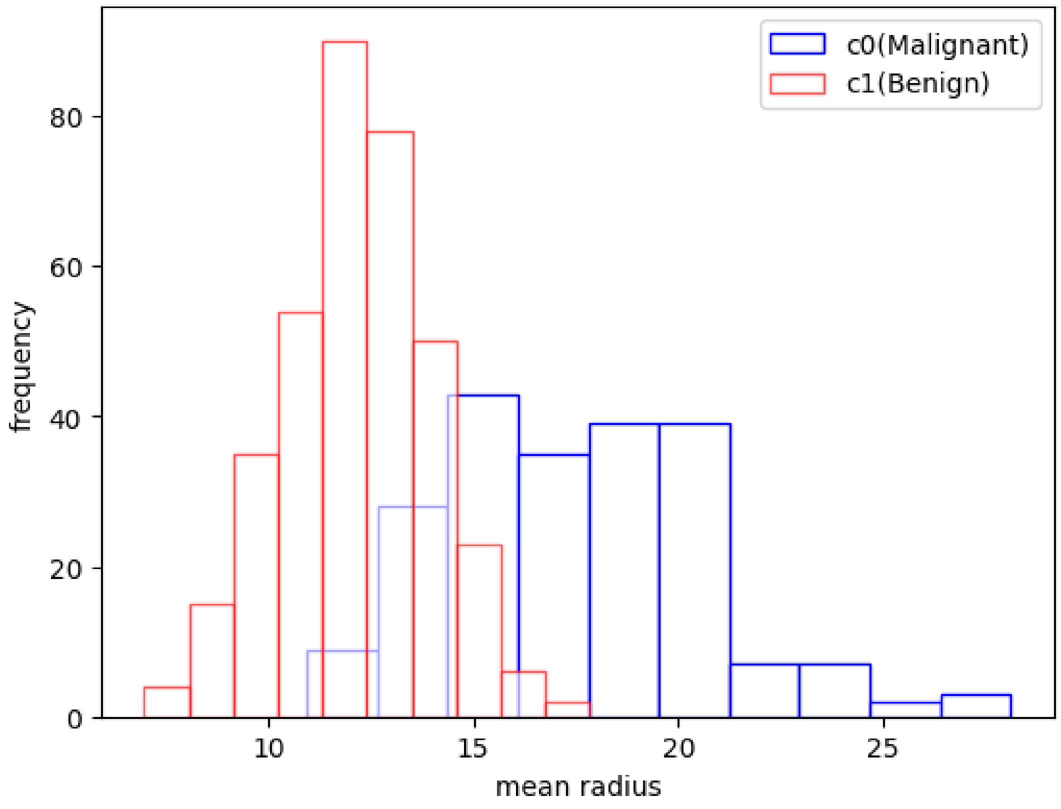
[5]:
```
df.iloc[[17, 18, 19, 20, 21], :] #Show row indices; 17, 18, 19, 20, 21
```

[5]:
```
    mean radius  mean perimeter  mean area  typeofcancer
17       16.130          108.10      798.8             0
18       19.810          130.00     1260.0             0
19       13.540           87.46      566.3             1
20       13.080           85.63      520.0             1
21        9.504           60.34      273.9             1
```

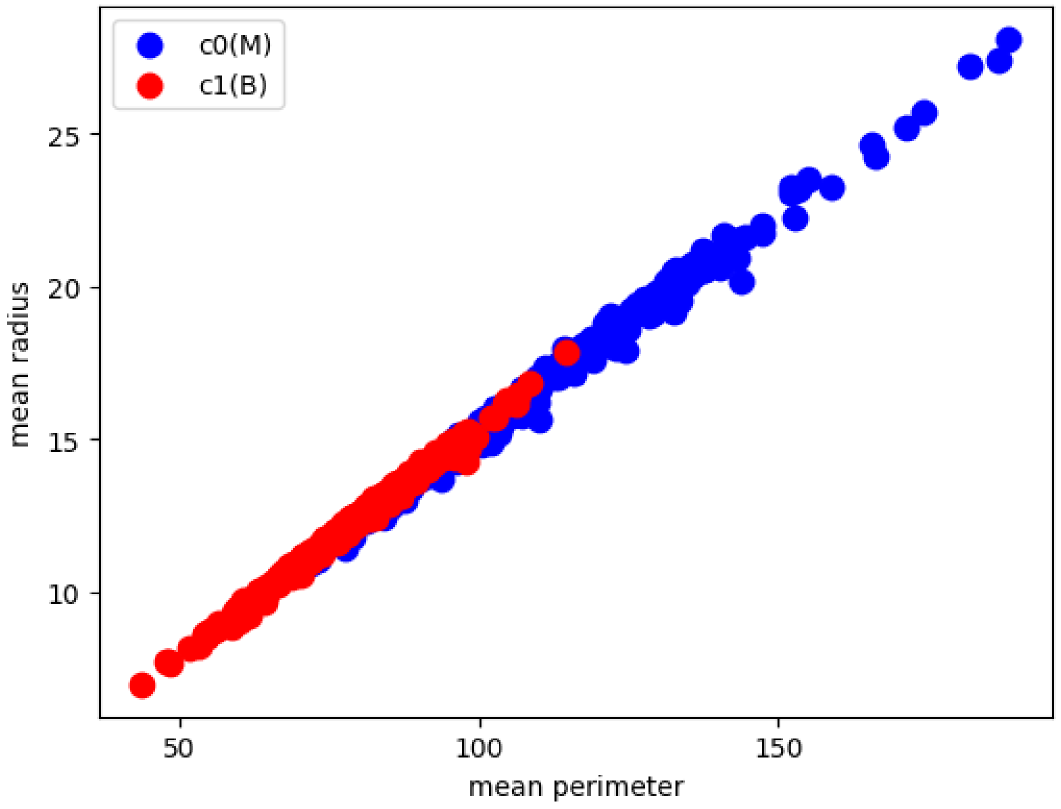
TASK 2

[6]:
```
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)')
```

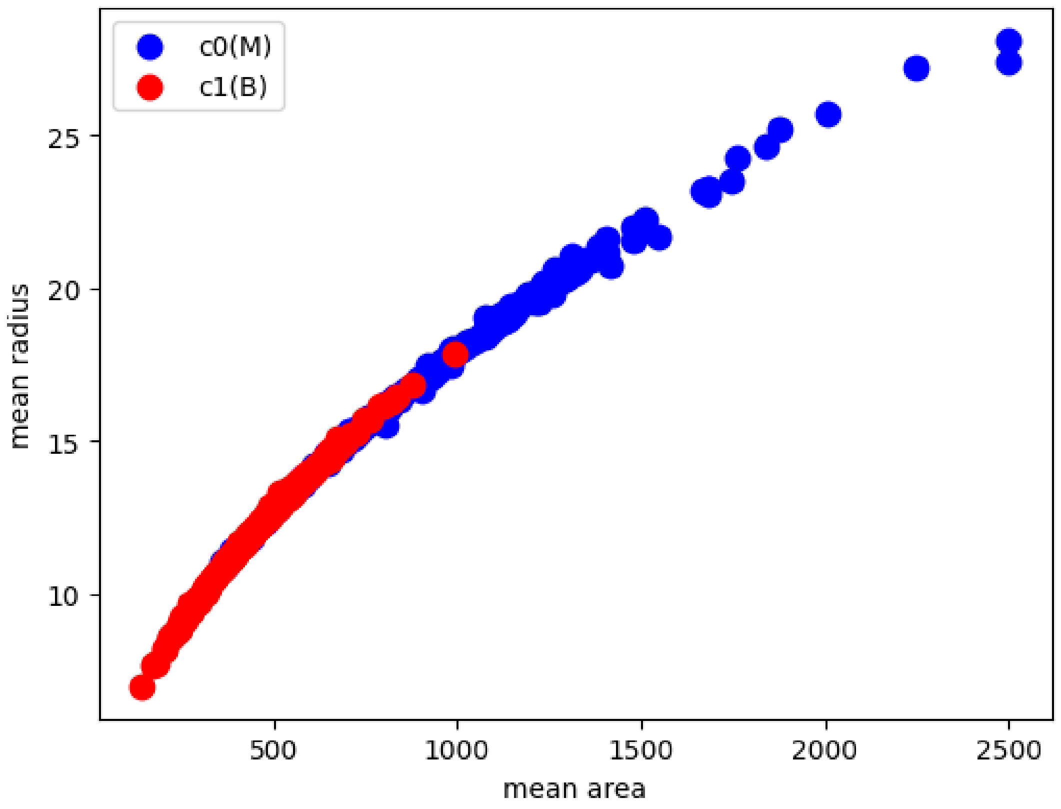
```
plt.hist(mean_radius_benign, bins=10, color = 'w', edgecolor='r', alpha=0.65,␣
 ↪label='c1(Benign)')


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



[7]:
```
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()
```



```
[8]: 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()
```


c0(M)
c1(B)
mean radius
mean area
25
20
15
10
500
1000
1500
2000
2500