04-advanced-classification

April 22, 2025

1 Topic 2: Advanced Classification Exercise

I'm following along the **Decision Trees** implementation from this YouTube.

This notebook can be accessed via GitHub repository here.

About the Dataset Dataset used in this exercise is taken from Kaggle

In this exercise, I'll be using Breast Cancer Wisconsin (Diagnostic) Data Set data.

Import necessary libraries & dataset

[21]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Load & display the dataset

[2]:
```
data = pd.read_csv("../../data/breast-cancer.csv")
data.head()
```

[2]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	\
0	842302	M	17.99	10.38	122.80	1001.0	
1	842517	M	20.57	17.77	132.90	1326.0	
2	84300903	M	19.69	21.25	130.00	1203.0	
3	84348301	M	11.42	20.38	77.58	386.1	
4	84358402	M	20.29	14.34	135.10	1297.0	

	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	\
0	0.11840	0.27760	0.3001	0.14710	
1	0.08474	0.07864	0.0869	0.07017	
2	0.10960	0.15990	0.1974	0.12790	
3	0.14250	0.28390	0.2414	0.10520	
4	0.10030	0.13280	0.1980	0.10430	

	…	texture_worst	perimeter_worst	area_worst	smoothness_worst	\
0	…	17.33	184.60	2019.0	0.1622	
1	…	23.41	158.80	1956.0	0.1238	
2	…	25.53	152.50	1709.0	0.1444	
3	…	26.50	98.87	567.7	0.2098	

```
4  …          16.67           152.20      1575.0           0.1374

   compactness_worst  concavity_worst  concave points_worst  symmetry_worst  \
0             0.6656           0.7119                0.2654          0.4601
1             0.1866           0.2416                0.1860          0.2750
2             0.4245           0.4504                0.2430          0.3613
3             0.8663           0.6869                0.2575          0.6638
4             0.2050           0.4000                0.1625          0.2364

   fractal_dimension_worst  Unnamed: 32
0                  0.11890          NaN
1                  0.08902          NaN
2                  0.08758          NaN
3                  0.17300          NaN
4                  0.07678          NaN

[5 rows x 33 columns]
```

Understanding the dataset

[3]:
```
data.info() # to get the information on the columns and datatype
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 33 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   id                       569 non-null    int64
 1   diagnosis                569 non-null    object
 2   radius_mean              569 non-null    float64
 3   texture_mean             569 non-null    float64
 4   perimeter_mean           569 non-null    float64
 5   area_mean                569 non-null    float64
 6   smoothness_mean          569 non-null    float64
 7   compactness_mean         569 non-null    float64
 8   concavity_mean           569 non-null    float64
 9   concave points_mean      569 non-null    float64
 10  symmetry_mean            569 non-null    float64
 11  fractal_dimension_mean   569 non-null    float64
 12  radius_se                569 non-null    float64
 13  texture_se               569 non-null    float64
 14  perimeter_se             569 non-null    float64
 15  area_se                  569 non-null    float64
 16  smoothness_se            569 non-null    float64
 17  compactness_se           569 non-null    float64
 18  concavity_se             569 non-null    float64
 19  concave points_se        569 non-null    float64
 20  symmetry_se              569 non-null    float64
```

```
 21  fractal_dimension_se     569 non-null    float64
 22  radius_worst             569 non-null    float64
 23  texture_worst            569 non-null    float64
 24  perimeter_worst          569 non-null    float64
 25  area_worst               569 non-null    float64
 26  smoothness_worst         569 non-null    float64
 27  compactness_worst        569 non-null    float64
 28  concavity_worst          569 non-null    float64
 29  concave points_worst     569 non-null    float64
 30  symmetry_worst           569 non-null    float64
 31  fractal_dimension_worst  569 non-null    float64
 32  Unnamed: 32              0 non-null      float64
dtypes: float64(31), int64(1), object(1)
memory usage: 146.8+ KB
```

[4]:
```
data.describe # to understand the numerical values
```

[4]:
```
<bound method NDFrame.describe of            id diagnosis  radius_mean
texture_mean  perimeter_mean  area_mean  \
0        842302         M        17.99         10.38          122.80     1001.0
1        842517         M        20.57         17.77          132.90     1326.0
2      84300903         M        19.69         21.25          130.00     1203.0
3      84348301         M        11.42         20.38           77.58      386.1
4      84358402         M        20.29         14.34          135.10     1297.0
..          …       …           …             …                …          …
564      926424         M        21.56         22.39          142.00     1479.0
565      926682         M        20.13         28.25          131.20     1261.0
566      926954         M        16.60         28.08          108.30      858.1
567      927241         M        20.60         29.33          140.10     1265.0
568       92751         B         7.76         24.54           47.92      181.0

     smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
0            0.11840           0.27760         0.30010              0.14710
1            0.08474           0.07864         0.08690              0.07017
2            0.10960           0.15990         0.19740              0.12790
3            0.14250           0.28390         0.24140              0.10520
4            0.10030           0.13280         0.19800              0.10430
..               …                 …               …                    …
564          0.11100           0.11590         0.24390              0.13890
565          0.09780           0.10340         0.14400              0.09791
566          0.08455           0.10230         0.09251              0.05302
567          0.11780           0.27700         0.35140              0.15200
568          0.05263           0.04362         0.00000              0.00000

     …  texture_worst  perimeter_worst  area_worst  smoothness_worst  \
0    …          17.33           184.60      2019.0           0.16220
1    …          23.41           158.80      1956.0           0.12380
```

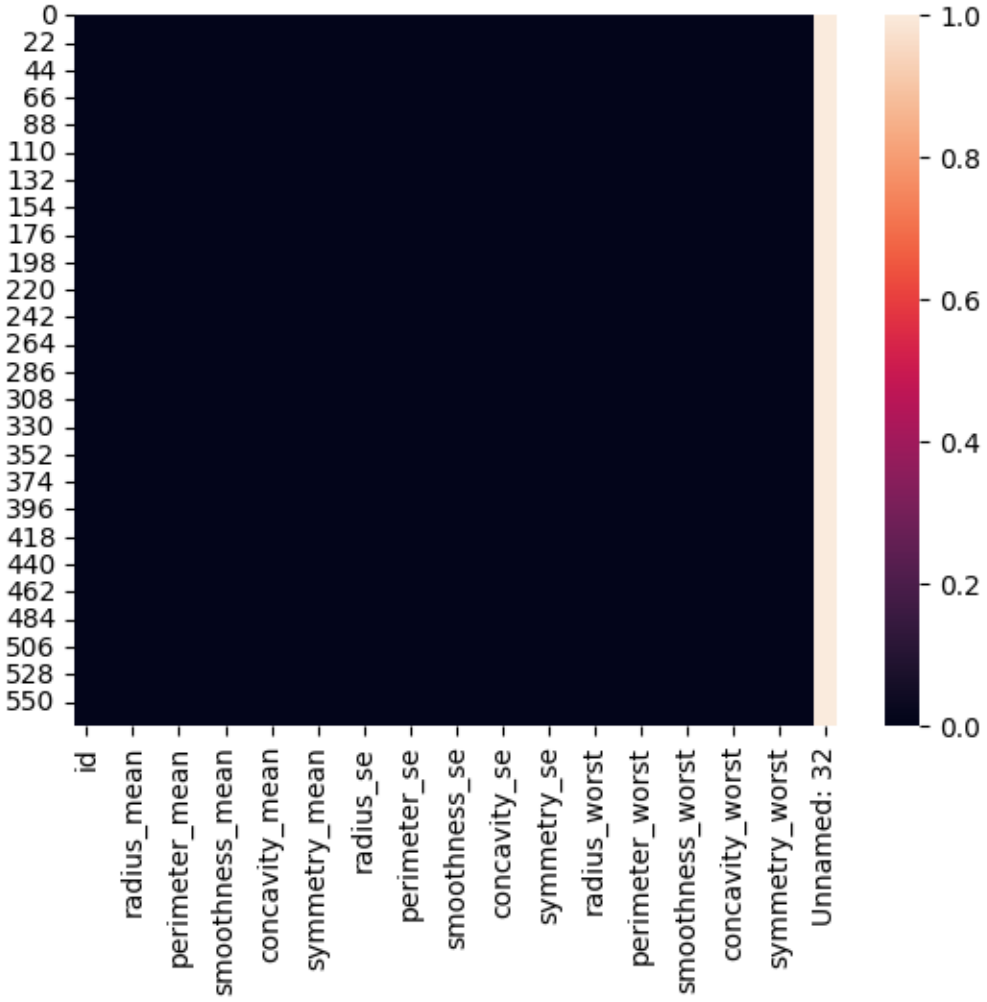
```
2    …          25.53          152.50      1709.0          0.14440
3    …          26.50           98.87       567.7          0.20980
4    …          16.67          152.20      1575.0          0.13740
..   …            …              …          …              …
564  …          26.40          166.10      2027.0          0.14100
565  …          38.25          155.00      1731.0          0.11660
566  …          34.12          126.70      1124.0          0.11390
567  …          39.42          184.60      1821.0          0.16500
568  …          30.37           59.16       268.6          0.08996

     compactness_worst  concavity_worst  concave points_worst  symmetry_worst  \
0              0.66560           0.7119                0.2654          0.4601
1              0.18660           0.2416                0.1860          0.2750
2              0.42450           0.4504                0.2430          0.3613
3              0.86630           0.6869                0.2575          0.6638
4              0.20500           0.4000                0.1625          0.2364
..                 …                …                     …               …
564            0.21130           0.4107                0.2216          0.2060
565            0.19220           0.3215                0.1628          0.2572
566            0.30940           0.3403                0.1418          0.2218
567            0.86810           0.9387                0.2650          0.4087
568            0.06444           0.0000                0.0000          0.2871

     fractal_dimension_worst  Unnamed: 32
0                    0.11890          NaN
1                    0.08902          NaN
2                    0.08758          NaN
3                    0.17300          NaN
4                    0.07678          NaN
..                       …            …
564                  0.07115          NaN
565                  0.06637          NaN
566                  0.07820          NaN
567                  0.12400          NaN
568                  0.07039          NaN

[569 rows x 33 columns]>
```

Cleaning the data

- Deal with the null value
  - use seaborn
- Deal with null column
  - Unnamed: 32
  - id

[5]:
```
sns.heatmap(data.isnull())
```

[5]: <Axes: >



[9]:
```
# drop the column

data.drop(['Unnamed: 32', 'id'], inplace=True, axis=1)
```

[10]:
```
data.head()
```

[10]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	\
0	M	17.99	10.38	122.80	1001.0	
1	M	20.57	17.77	132.90	1326.0	
2	M	19.69	21.25	130.00	1203.0	
3	M	11.42	20.38	77.58	386.1	
4	M	20.29	14.34	135.10	1297.0	

```
   smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
0          0.11840           0.27760          0.3001              0.14710
1          0.08474           0.07864          0.0869              0.07017
2          0.10960           0.15990          0.1974              0.12790
3          0.14250           0.28390          0.2414              0.10520
4          0.10030           0.13280          0.1980              0.10430

   symmetry_mean  …  radius_worst  texture_worst  perimeter_worst  \
0         0.2419  …         25.38          17.33           184.60
1         0.1812  …         24.99          23.41           158.80
2         0.2069  …         23.57          25.53           152.50
3         0.2597  …         14.91          26.50            98.87
4         0.1809  …         22.54          16.67           152.20

   area_worst  smoothness_worst  compactness_worst  concavity_worst  \
0      2019.0            0.1622             0.6656           0.7119
1      1956.0            0.1238             0.1866           0.2416
2      1709.0            0.1444             0.4245           0.4504
3       567.7            0.2098             0.8663           0.6869
4      1575.0            0.1374             0.2050           0.4000

   concave points_worst  symmetry_worst  fractal_dimension_worst
0                0.2654          0.4601                  0.11890
1                0.1860          0.2750                  0.08902
2                0.2430          0.3613                  0.08758
3                0.2575          0.6638                  0.17300
4                0.1625          0.2364                  0.07678

[5 rows x 31 columns]
```

Change diagnosis class into numerical value (0 and 1)

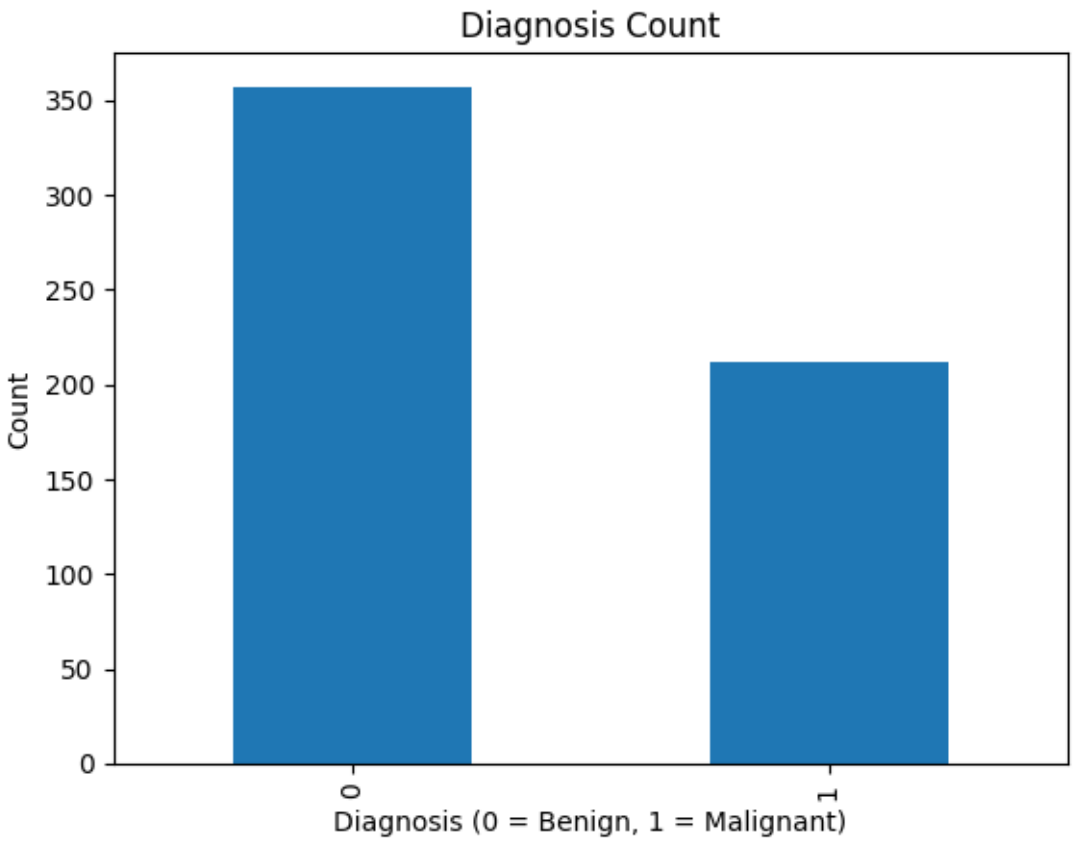
[ ]:
```
from sklearn.preprocessing import LabelEncoder

encoder = LabelEncoder()
data['diagnosis'] = encoder.fit_transform(data['diagnosis'])
```

[13]:
```
data.head()
```

[13]:
```
   diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
0          1        17.99         10.38          122.80     1001.0
1          1        20.57         17.77          132.90     1326.0
2          1        19.69         21.25          130.00     1203.0
3          1        11.42         20.38           77.58      386.1
4          1        20.29         14.34          135.10     1297.0

   smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
0          0.11840           0.27760          0.3001              0.14710
```

```
1          0.08474           0.07864          0.0869             0.07017
2          0.10960           0.15990          0.1974             0.12790
3          0.14250           0.28390          0.2414             0.10520
4          0.10030           0.13280          0.1980             0.10430

   symmetry_mean  …  radius_worst  texture_worst  perimeter_worst  \
0         0.2419  …         25.38          17.33           184.60
1         0.1812  …         24.99          23.41           158.80
2         0.2069  …         23.57          25.53           152.50
3         0.2597  …         14.91          26.50            98.87
4         0.1809  …         22.54          16.67           152.20

   area_worst  smoothness_worst  compactness_worst  concavity_worst  \
0      2019.0            0.1622             0.6656           0.7119
1      1956.0            0.1238             0.1866           0.2416
2      1709.0            0.1444             0.4245           0.4504
3       567.7            0.2098             0.8663           0.6869
4      1575.0            0.1374             0.2050           0.4000

   concave points_worst  symmetry_worst  fractal_dimension_worst
0                0.2654          0.4601                  0.11890
1                0.1860          0.2750                  0.08902
2                0.2430          0.3613                  0.08758
3                0.2575          0.6638                  0.17300
4                0.1625          0.2364                  0.07678

[5 rows x 31 columns]
```

[26]:
```
# simple eda for quick view

data["diagnosis"].value_counts().plot(kind='bar')
plt.title("Diagnosis Count")
plt.xlabel("Diagnosis (0 = Benign, 1 = Malignant)")
plt.ylabel("Count")
plt.show()
```



Dividing the predictors and the target variable We also want to normalize the data to ensure that all features contribute equally to the model by bringing them to a similar scale

[28]:
```
y = data["diagnosis"] # target variable
X = data.drop(["diagnosis"], axis=1) # we also can use .iloc here just like Dr␣
↪Syahid instructed
```

[ ]:
```
from sklearn.preprocessing import StandardScaler

# create a scaler object

scaler = StandardScaler()

# fit the scaler to the data and transform the data

X_scaled = scaler.fit_transform(X)
```

[ ]:
```
   radius_mean  texture_mean  perimeter_mean  area_mean  smoothness_mean  \
0        17.99         10.38          122.80     1001.0          0.11840
1        20.57         17.77          132.90     1326.0          0.08474
```

```
2          19.69         21.25          130.00     1203.0          0.10960
3          11.42         20.38           77.58      386.1          0.14250
4          20.29         14.34          135.10     1297.0          0.10030
..           …             …               …          …               …
564        21.56         22.39          142.00     1479.0          0.11100
565        20.13         28.25          131.20     1261.0          0.09780
566        16.60         28.08          108.30      858.1          0.08455
567        20.60         29.33          140.10     1265.0          0.11780
568         7.76         24.54           47.92      181.0          0.05263

     compactness_mean  concavity_mean  concave points_mean  symmetry_mean  \
0             0.27760         0.30010              0.14710         0.2419
1             0.07864         0.08690              0.07017         0.1812
2             0.15990         0.19740              0.12790         0.2069
3             0.28390         0.24140              0.10520         0.2597
4             0.13280         0.19800              0.10430         0.1809
..                …               …                    …            …
564           0.11590         0.24390              0.13890         0.1726
565           0.10340         0.14400              0.09791         0.1752
566           0.10230         0.09251              0.05302         0.1590
567           0.27700         0.35140              0.15200         0.2397
568           0.04362         0.00000              0.00000         0.1587

     fractal_dimension_mean  …  radius_worst  texture_worst  \
0                   0.07871  …        25.380          17.33
1                   0.05667  …        24.990          23.41
2                   0.05999  …        23.570          25.53
3                   0.09744  …        14.910          26.50
4                   0.05883  …        22.540          16.67
..                      …  …           …               …
564                 0.05623  …        25.450          26.40
565                 0.05533  …        23.690          38.25
566                 0.05648  …        18.980          34.12
567                 0.07016  …        25.740          39.42
568                 0.05884  …         9.456          30.37

     perimeter_worst  area_worst  smoothness_worst  compactness_worst  \
0             184.60      2019.0           0.16220            0.66560
1             158.80      1956.0           0.12380            0.18660
2             152.50      1709.0           0.14440            0.42450
3              98.87       567.7           0.20980            0.86630
4             152.20      1575.0           0.13740            0.20500
..               …           …                …                  …
564           166.10      2027.0           0.14100            0.21130
565           155.00      1731.0           0.11660            0.19220
566           126.70      1124.0           0.11390            0.30940
567           184.60      1821.0           0.16500            0.86810
```

```
568            59.16       268.6           0.08996            0.06444

     concavity_worst  concave points_worst  symmetry_worst  \
0             0.7119                0.2654          0.4601
1             0.2416                0.1860          0.2750
2             0.4504                0.2430          0.3613
3             0.6869                0.2575          0.6638
4             0.4000                0.1625          0.2364
..               …                     …               …
564           0.4107                0.2216          0.2060
565           0.3215                0.1628          0.2572
566           0.3403                0.1418          0.2218
567           0.9387                0.2650          0.4087
568           0.0000                0.0000          0.2871

     fractal_dimension_worst
0                    0.11890
1                    0.08902
2                    0.08758
3                    0.17300
4                    0.07678
..                       …
564                  0.07115
565                  0.06637
566                  0.07820
567                  0.12400
568                  0.07039

[569 rows x 30 columns]
```

[31]:
```
X_scaled
```

[31]:
```
array([[ 1.09706398, -2.07333501,  1.26993369, …,  2.29607613,
         2.75062224,  1.93701461],
       [ 1.82982061, -0.35363241,  1.68595471, …,  1.0870843 ,
        -0.24388967,  0.28118999],
       [ 1.57988811,  0.45618695,  1.56650313, …,  1.95500035,
         1.152255  ,  0.20139121],
       …,
       [ 0.70228425,  2.0455738 ,  0.67267578, …,  0.41406869,
        -1.10454895, -0.31840916],
       [ 1.83834103,  2.33645719,  1.98252415, …,  2.28998549,
         1.91908301,  2.21963528],
       [-1.80840125,  1.22179204, -1.81438851, …, -1.74506282,
        -0.04813821, -0.75120669]], shape=(569, 30))
```

Split the data into training and testing sets

[33]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3,␣
↪random_state=42)
```

Train the model

[34]:
```
from sklearn.linear_model import LogisticRegression

# create the lr model

lr = LogisticRegression()

# train the model on the training data

lr.fit(X_train, y_train)

# predict the target variable based on the test data

y_pred = lr.predict(X_test)
```

Evaluating the model performance

[39]:
```
#accuracy score

from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred)
```

[39]: 0.9824561403508771

[40]:
```
# precision score

from sklearn.metrics import precision_score
precision_score(y_test, y_pred)
```

[40]: 0.96875

[42]:
```
# recall

from sklearn.metrics import recall_score
recall_score(y_test, y_pred)
```

[42]: 0.9841269841269841

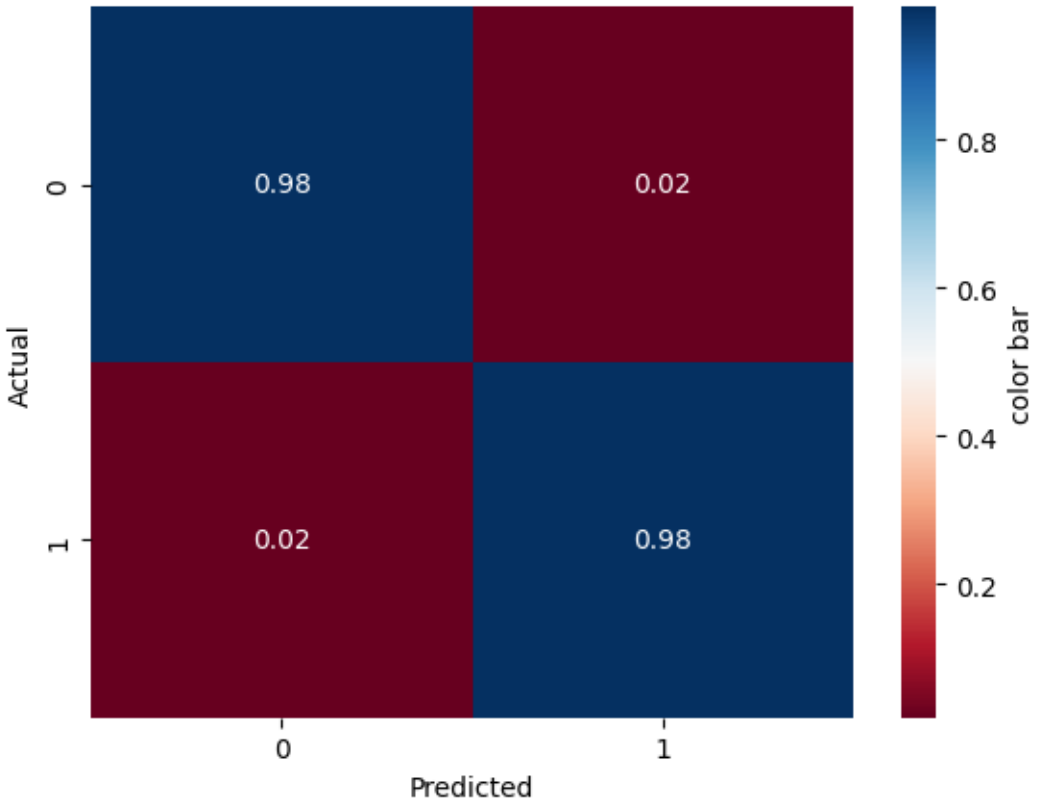
[43]:
```
# classification report (this is an extensive report from the library)

from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred, target_names␣
↪=['malignant','benign']))
```

```
              precision    recall  f1-score   support

   malignant       0.99      0.98      0.99       108
      benign       0.97      0.98      0.98        63

    accuracy                           0.98       171
   macro avg       0.98      0.98      0.98       171
weighted avg       0.98      0.98      0.98       171
```

[44]:
```
# confusion matrix

from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred, labels=[0,1])
cm
```

[44]:
```
array([[106,   2],
       [  1,  62]])
```

[45]:
```
# Let's visualize a heatmap for our confusion matrix using seaborn & matplotlib
# We also want to normalize the numbers to be between -1 to 1 using numpy

import numpy as np

cm_normalized = np.round(cm/np.sum(cm, axis=1).reshape(-1,1),2)

import seaborn as sns
import matplotlib.pyplot as plt
sns.heatmap(cm_normalized, cmap="RdBu", annot=True,
            cbar_kws={"orientation":"vertical","label":"color bar"},
            xticklabels=[0,1], yticklabels=[0,1]
            )
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```


0.98
0.02
0.02
0.98
Actual
0
1
0
1
Predicted
0.8
0.6
0.4
0.2
color bar