## HW3

## October 14, 2023

# 0.1 # Homework #3 Student Name: Sam Crane

Student ID: 801101091

GitHub: https://github.com/samofuture/Intro-to-ML

```
[]: import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

## 0.2 Problem 1

```
[]: sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

```
[]: #Import LogisticRegression from sklearn.linear_model

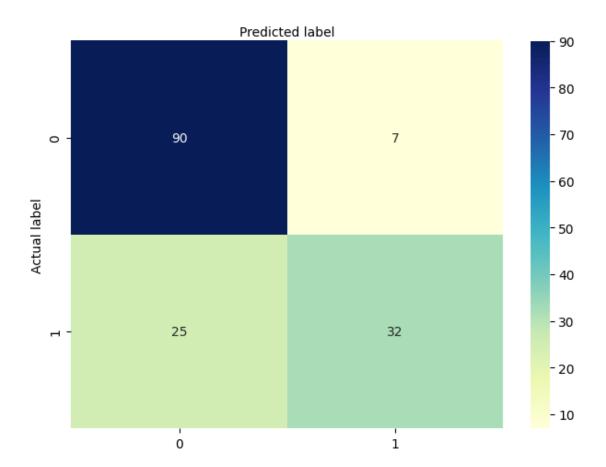
#Make an instance classifier of the object LogisticRegression and give_

\( \to random_state = 0 \)

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score
```

```
classifier = LogisticRegression(random_state=10)
     classifier.fit(X_train, Y_train)
[]: LogisticRegression(random_state=10)
[]: #Using Confusion matrix we can get accuracy of our model.
     from sklearn.metrics import confusion_matrix
     Y_pred = classifier.predict(X_test)
     cnf_matrix = confusion_matrix(Y_test, Y_pred)
     cnf_matrix
[]: array([[90, 7],
            [25, 32]])
[]: #Let's evaluate the model using model evaluation metrics such as accuracy, __
     ⇔precision, and recall.
     from sklearn import metrics
     print(metrics.classification_report(Y_test, Y_pred))
                  precision
                               recall f1-score
                                                   support
               0
                       0.78
                                 0.93
                                           0.85
                                                        97
                       0.82
                                 0.56
                                           0.67
                                                        57
                                           0.79
                                                       154
        accuracy
                                           0.76
       macro avg
                       0.80
                                 0.74
                                                       154
                       0.80
                                 0.79
                                           0.78
                                                       154
    weighted avg
[]: #Let's visualize the results of the model in the form of a co#nfusion matrix.
     ⇔using matplotlib and seaborn.
     #Here, you will visualize the confusion matrix using Heatmap.
     import seaborn as sns
     class_names=[0,1] # name of classes
     fig, ax = plt.subplots()
     tick_marks = np.arange(len(class_names))
     plt.xticks(tick_marks, class_names)
     plt.yticks(tick_marks, class_names)
     # create heatmap
     sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu" ,fmt='g')
     ax.xaxis.set_label_position("top")
     plt.tight_layout()
     plt.title('Confusion matrix', y=1.1)
     plt.ylabel('Actual label')
     plt.xlabel('Predicted label')
```

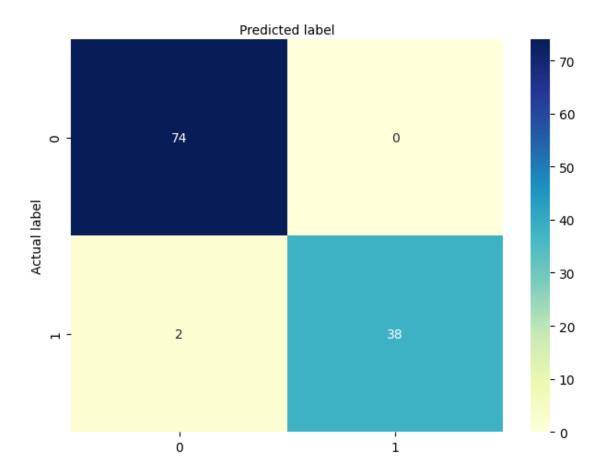


## 0.3 Problem 2a

```
[]: df = pd.read_csv('cancer.csv')
df['diagnosis'] = df['diagnosis'].apply(lambda x: 1 if x == 'M' else 0)
Y = df.iloc[:, 1].values
df = df.drop(columns='diagnosis', axis=1)
X = df.iloc[:, 1:].values

#Now we'll split our Data set into Training Data and Test Data. Training data_
will be used to train our
#Logistic model and Test data will be used to validate our model. We'll use_
Sklearn to split our data. We'll import train_test_split from sklearn.
model_selection
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, \( \text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\tex{
```

```
[]: sc_X = StandardScaler()
     X_train = sc_X.fit_transform(X_train)
     X_test = sc_X.transform(X_test)
[]: classifier = LogisticRegression(random_state=10)
     classifier.fit(X_train, Y_train)
[]: LogisticRegression(random_state=10)
[]: #Using Confusion matrix we can get accuracy of our model.
     Y_pred = classifier.predict(X_test)
     cnf_matrix = confusion_matrix(Y_test, Y_pred)
[]: print(metrics.classification_report(Y_test, Y_pred))
                  precision
                               recall f1-score
                                                  support
               0
                       0.97
                                 1.00
                                           0.99
                                                        74
                       1.00
                                 0.95
                                           0.97
               1
                                                        40
                                           0.98
                                                       114
        accuracy
                                           0.98
                                                       114
       macro avg
                       0.99
                                 0.97
    weighted avg
                       0.98
                                 0.98
                                           0.98
                                                       114
[]: class_names=[0,1] # name of classes
     fig, ax = plt.subplots()
     tick_marks = np.arange(len(class_names))
     plt.xticks(tick_marks, class_names)
     plt.yticks(tick_marks, class_names)
     # create heatmap
     sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu",fmt='g')
     ax.xaxis.set_label_position("top")
     plt.tight_layout()
     plt.title('Confusion matrix', y=1.1)
     plt.ylabel('Actual label')
     plt.xlabel('Predicted label')
```



## 0.4 Problem 2b

```
[]: classifier = LogisticRegression(random_state=10, penalty='12', C=0.8)
    classifier.fit(X_train, Y_train)
    Y_pred = classifier.predict(X_test)
    cnf_matrix = confusion_matrix(Y_test, Y_pred)

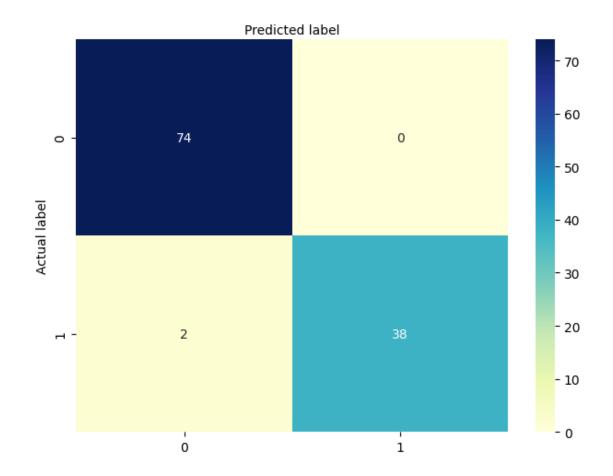
print(metrics.classification_report(Y_test, Y_pred))

class_names=[0,1] # name of classes
fig, ax = plt.subplots()
    tick_marks = np.arange(len(class_names))
    plt.xticks(tick_marks, class_names)
    plt.yticks(tick_marks, class_names)
# create heatmap
sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu",fmt='g')
ax.xaxis.set_label_position("top")
```

```
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
```

	precision	recall	f1-score	support
0 1	0.97 1.00	1.00 0.95	0.99 0.97	74 40
accuracy			0.98	114
macro avg	0.99	0.97	0.98	114
weighted avg	0.98	0.98	0.98	114

[]: Text(0.5, 427.9555555555555, 'Predicted label')



```
0.5 Problem 3
[]: df = pd.read_csv('cancer.csv')
     df['diagnosis'] = df['diagnosis'].apply(lambda x: 1 if x == 'M' else 0)
     Y = df.iloc[:, 1].values
     df = df.drop(columns='diagnosis', axis=1)
     X = df.iloc[:, 1:].values
     #Now we'll split our Data set into Training Data and Test Data. Training data_
      ⇔will be used to train our
     #Logistic model and Test data will be used to validate our model. We'll use,
      Sklearn to split our data. We'll import train_test_split from sklearn.
     \hookrightarrow model_selection
     X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, ___
      →random_state = 7)
[]: sc_X = StandardScaler()
     X_train = sc_X.fit_transform(X_train)
     X_test = sc_X.transform(X_test)
[]: from sklearn.naive_bayes import GaussianNB
     # fit a Naive Bayes model to the data
     model = GaussianNB()
     model.fit(X_train, Y_train)
     # make predictions
```

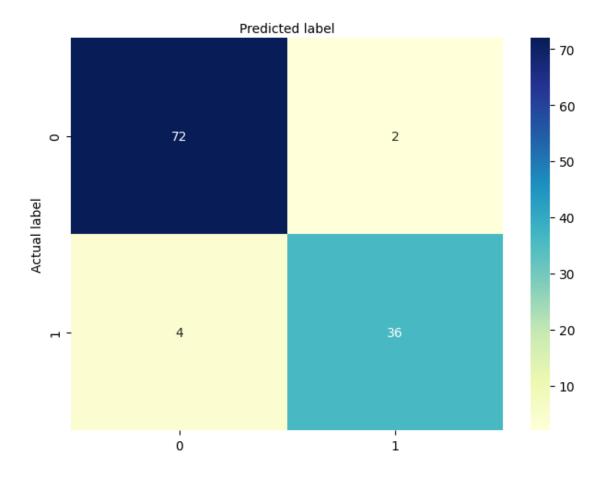
```
expected = Y_test
predicted = model.predict(X_test)
# summarize the fit of the model
print(metrics.classification report(expected, predicted))
```

	precision	recall	f1-score	support
0	0.95	0.97	0.96	74
1	0.95	0.90	0.92	40
accuracy			0.95	114
macro avg	0.95	0.94	0.94	114
weighted avg	0.95	0.95	0.95	114

```
[]: cnf_matrix = metrics.confusion_matrix(expected, predicted)
     class_names=[0,1] # name of classes
     fig, ax = plt.subplots()
     tick_marks = np.arange(len(class_names))
     plt.xticks(tick_marks, class_names)
     plt.yticks(tick_marks, class_names)
```

```
# create heatmap
sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu" ,fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
```

[]: Text(0.5, 427.9555555555555, 'Predicted label')



The Bayesian classifier performed worse than the normal Logistic Regression in problem 2a. Although, it outperformed Problem 2b in recall and precision. These conclusions are also supported by the f1 score, where 2a has the highest, 3 is second highest, and 2b is lowest.

### 0.6 Problem 4

```
[]: df = pd.read_csv('cancer.csv')
     df['diagnosis'] = df['diagnosis'].apply(lambda x: 1 if x == 'M' else 0)
     Y = df.iloc[:, 1].values
     df = df.drop(columns='diagnosis', axis=1)
     X = df.iloc[:, 1:].values
[]: from sklearn.decomposition import PCA
    k = 30
    \max \ \text{accuracy} = 0
     \max acc index = 0
     for n in range(1, k+1):
         pc_list = [f'pc{i}' for i in range(n)]
         pca = PCA(n_components=n)
         principalComponents = pca.fit_transform(X)
         principalDf = pd.DataFrame(data = principalComponents
                     , columns = pc_list)
         X_train, X_test, Y_train, Y_test = train_test_split(principalDf, Y,__
      stest_size = 0.2, random_state = 7)
         sc_X = StandardScaler()
         X_train = sc_X.fit_transform(X_train)
         X_test = sc_X.transform(X_test)
         classifier = LogisticRegression(random_state=10)
         classifier.fit(X_train, Y_train)
         Y_pred = classifier.predict(X_test)
         cnf_matrix = confusion_matrix(Y_test, Y_pred)
         acc = metrics.accuracy_score(Y_test, Y_pred)
         if acc > max_accuracy:
             max_accuracy = acc
             max_acc_index = n
         # print(f'{n} PC:', acc)
         print(metrics.classification_report(Y_test, Y_pred))
     print(f'{max_acc_index}: {max_accuracy}')
```

	precision	recall	f1-score	support
0 1	0.90 0.97	0.99 0.80	0.94 0.88	74 40
accuracy macro avg weighted avg	0.94 0.93	0.89 0.92	0.92 0.91 0.92	114 114 114
	precision	recall	f1-score	support

•	0.00	4 00	0.04	7.4
0	0.89	1.00	0.94	74
1	1.00	0.78	0.87	40
accuracy			0.92	114
•	0.05	0.00		
macro avg	0.95	0.89	0.91	114
weighted avg	0.93	0.92	0.92	114
	precision	recall	f1-score	support
	F			
^	0.00	4 00	0.04	7.4
0	0.89	1.00	0.94	74
1	1.00	0.78	0.87	40
accuracy			0.92	114
macro avg	0.95	0.89	0.91	114
_				
weighted avg	0.93	0.92	0.92	114
	precision	recall	f1-score	support
	_			
0	0.91	0.99	0.95	74
1	0.97	0.82	0.89	40
accuracy			0.93	114
macro avg	0.94	0.91	0.92	114
weighted avg	0.93	0.93	0.93	114
weighted avg	0.93	0.93	0.93	114
	precision	recall	f1-score	support
0	0.93	1.00	0.96	74
1	1.00	0.85	0.92	40
1	1.00	0.05	0.92	40
accuracy			0.95	114
macro avg	0.96	0.93	0.94	114
weighted avg	0.95	0.95	0.95	114
0				
			C.4	
	precision	recall	f1-score	support
0	0.93	1.00	0.96	74
1	1.00	0.85	0.92	40
_				
			0.05	444
accuracy			0.95	114
macro avg	0.96	0.93	0.94	114
weighted avg	0.95	0.95	0.95	114
5 0				
	precision	recall	f1-score	gunnor+
	brecretom	recarr	II PCOLE	support
0	0.93	1.00	0.96	74
1	1.00	0.85	0.92	40

accuracy			0.95	114
macro avg	0.96	0.93	0.94	114
_				
weighted avg	0.95	0.95	0.95	114
	precision	recall	f1-score	support
	precision	recarr	II SCOLE	Suppor c
0	0.93	1.00	0.96	74
1	1.00	0.85	0.92	40
Τ.	1.00	0.05	0.92	40
accuracy			0.95	114
•	0.00	0.00		
macro avg	0.96	0.93	0.94	114
weighted avg	0.95	0.95	0.95	114
o o				
	precision	recall	f1-score	support
0	0.93	1.00	0.96	74
1	1.00	0.85	0.92	40
			0.05	444
accuracy			0.95	114
macro avg	0.96	0.93	0.94	114
weighted avg	0.95	0.95	0.95	114
weighted avg	0.30	0.50	0.30	11-1
	precision	recall	f1-score	support
	1			11
0	0.91	1.00	0.95	74
1	1.00	0.82	0.90	40
=	1.00	0.02	0.00	10
accuracy			0.94	114
macro avg	0.96	0.91	0.93	114
•				
weighted avg	0.94	0.94	0.94	114
	precision	recall	f1-gcore	gunnort
	precision	recall	f1-score	support
	precision	recall	f1-score	support
0	-			
0	0.91	1.00	0.95	74
0 1	-			
	0.91	1.00	0.95	74
1	0.91	1.00	0.95 0.90	74 40
1 accuracy	0.91 1.00	1.00 0.82	0.95 0.90 0.94	74 40 114
accuracy macro avg	0.91 1.00 0.96	1.00 0.82	0.95 0.90 0.94 0.93	74 40 114 114
1 accuracy	0.91 1.00	1.00 0.82	0.95 0.90 0.94	74 40 114
accuracy macro avg	0.91 1.00 0.96	1.00 0.82	0.95 0.90 0.94 0.93	74 40 114 114
accuracy macro avg	0.91 1.00 0.96 0.94	1.00 0.82 0.91 0.94	0.95 0.90 0.94 0.93 0.94	74 40 114 114 114
accuracy macro avg	0.91 1.00 0.96	1.00 0.82	0.95 0.90 0.94 0.93	74 40 114 114
accuracy macro avg	0.91 1.00 0.96 0.94	1.00 0.82 0.91 0.94	0.95 0.90 0.94 0.93 0.94	74 40 114 114 114
accuracy macro avg weighted avg	0.91 1.00 0.96 0.94 precision	1.00 0.82 0.91 0.94 recall	0.95 0.90 0.94 0.93 0.94 f1-score	74 40 114 114 114 support
accuracy macro avg weighted avg	0.91 1.00 0.96 0.94 precision 0.91	1.00 0.82 0.91 0.94 recall	0.95 0.90 0.94 0.93 0.94 f1-score	74 40 114 114 114 support
accuracy macro avg weighted avg	0.91 1.00 0.96 0.94 precision	1.00 0.82 0.91 0.94 recall	0.95 0.90 0.94 0.93 0.94 f1-score	74 40 114 114 114 support
accuracy macro avg weighted avg	0.91 1.00 0.96 0.94 precision 0.91	1.00 0.82 0.91 0.94 recall	0.95 0.90 0.94 0.93 0.94 f1-score	74 40 114 114 114 support
accuracy macro avg weighted avg  0 1	0.91 1.00 0.96 0.94 precision 0.91	1.00 0.82 0.91 0.94 recall	0.95 0.90 0.94 0.93 0.94 f1-score 0.95 0.90	74 40 114 114 114 support 74 40
accuracy macro avg weighted avg  0 1 accuracy	0.91 1.00 0.96 0.94 precision 0.91 1.00	1.00 0.82 0.91 0.94 recall 1.00 0.82	0.95 0.90 0.94 0.93 0.94 f1-score 0.95 0.90	74 40 114 114 114 support 74 40
accuracy macro avg weighted avg  0 1	0.91 1.00 0.96 0.94 precision 0.91	1.00 0.82 0.91 0.94 recall	0.95 0.90 0.94 0.93 0.94 f1-score 0.95 0.90	74 40 114 114 114 support 74 40

weighted avg	0.94	0.94	0.94	114
	precision	recall	f1-score	support
0	0.01	1 00	0.05	71
0	0.91	1.00	0.95	74
1	1.00	0.82	0.90	40
accuracy			0.94	114
macro avg	0.96	0.91	0.93	114
weighted avg	0.94	0.94	0.94	114
	precision	recall	f1-score	support
0	0.91	1.00	0.95	74
1	1.00	0.82	0.90	40
accuracy			0.94	114
macro avg	0.96	0.91	0.93	114
weighted avg	0.94	0.94	0.94	114
	precision	recall	f1-score	support
0	0.91	1.00	0.95	74
1	1.00	0.82	0.90	40
accuracy			0.94	114
macro avg	0.96	0.91	0.93	114
weighted avg	0.94	0.94	0.94	114
8 44 4 8				
	precision	recall	f1-score	support
0	0.94	0.99	0.96	74
1	0.97	0.88	0.92	40
_	0.07	0.00	0.02	10
accuracy			0.95	114
macro avg	0.95	0.93	0.94	114
weighted avg		0.95	0.95	
	precision	recall	f1-score	support
0	0.94	1.00	0.97	74
1	1.00	0.88	0.93	40
_		2.00	2.00	10
accuracy			0.96	114
macro avg	0.97	0.94	0.95	114
weighted avg	0.96	0.96		
	precision	recall	f1-score	support

0	0.94	1.00	0.97	74
1	1.00	0.88	0.93	40
accuracy			0.96	114
macro avg	0.97	0.94	0.95	114
weighted avg	0.96	0.96	0.96	114
8				
	precision	recall	f1-score	support
0	0.93	1.00	0.96	74
1	1.00	0.85	0.92	40
_		0.00	0.02	
accuracy			0.95	114
macro avg	0.96	0.93	0.94	114
_				
weighted avg	0.95	0.95	0.95	114
	precision	recall	f1-score	support
0	0.05	1 00	0.07	7.1
0	0.95	1.00	0.97	74
1	1.00	0.90	0.95	40
2.661172.617			0.96	114
accuracy	0.07	0.05		
macro avg	0.97	0.95	0.96	114
weighted avg	0.97	0.96	0.96	114
	precision	recall	f1-score	support
0	0.94	1.00	0.97	74
1	1.00	0.88	0.93	40
1	1.00	0.00	0.93	40
accuracy			0.96	114
macro avg	0.97	0.94	0.95	114
_		0.94	0.96	
weighted avg	0.96	0.96	0.96	114
	precision	recall	f1-score	support
0	0.96	1.00	0.98	74
1	1.00	0.93	0.96	40
-	1.00	0.00	0.50	10
accuracy			0.97	114
macro avg	0.98	0.96	0.97	114
weighted avg	0.97	0.97	0.97	114
weighted avg	0.31	0.31	0.31	114
	precision	recall	f1-score	support
0	0.94	1.00	0.97	74
1	1.00	0.88	0.93	40
Τ.	1.00	0.00	0.33	40

accuracy			0.96	114
macro avg	0.97	0.94	0.95	114
weighted avg	0.96	0.96	0.96	114
weighted avg	0.90	0.90	0.90	114
	precision	recall	f1-score	support
	•			
0	0.94	1.00	0.07	74
0			0.97	
1	1.00	0.88	0.93	40
accuracy			0.96	114
•	0.07			
macro avg	0.97	0.94	0.95	114
weighted avg	0.96	0.96	0.96	114
0 0				
		7 7	£4	
	precision	recall	f1-score	support
0	0.94	1.00	0.97	74
	1.00	0.88	0.93	
1	1.00	0.88	0.93	40
accuracy			0.96	114
macro avg	0.97	0.94	0.95	114
•				
weighted avg	0.96	0.96	0.96	114
	precision	recall	f1-score	support
	P-00-0-0-1			zappoz c
0	0.94	1.00	0.97	74
1	1.00	0.88	0.93	40
			0.00	444
accuracy			0.96	114
macro avg	0.97	0.94	0.95	114
weighted avg				
	0.96	0.96	0.96	114
worghood avg	0.96	0.96	0.96	114
worghood dvg		0.96		114
working and	0.96 precision	0.96		114 support
worghood dvg				
	precision	recall	f1-score	support
0	precision 0.94	recall	f1-score 0.97	support
	precision	recall	f1-score	support
0	precision 0.94	recall	f1-score 0.97	support
0 1	precision 0.94	recall	f1-score 0.97 0.93	support 74 40
0 1 accuracy	precision 0.94 1.00	recall 1.00 0.88	f1-score 0.97 0.93 0.96	support 74 40 114
0 1 accuracy macro avg	0.94 1.00	1.00 0.88	f1-score 0.97 0.93 0.96 0.95	54 40 114 114
0 1 accuracy	precision 0.94 1.00	recall 1.00 0.88	f1-score 0.97 0.93 0.96	support 74 40 114
0 1 accuracy macro avg	0.94 1.00	1.00 0.88	f1-score 0.97 0.93 0.96 0.95	54 40 114 114
0 1 accuracy macro avg	0.94 1.00 0.97 0.96	1.00 0.88 0.94 0.96	f1-score 0.97 0.93 0.96 0.95 0.96	74 40 114 114 114
0 1 accuracy macro avg	0.94 1.00	1.00 0.88	f1-score 0.97 0.93 0.96 0.95	54 40 114 114
0 1 accuracy macro avg	0.94 1.00 0.97 0.96	1.00 0.88 0.94 0.96	f1-score 0.97 0.93 0.96 0.95 0.96	74 40 114 114 114
0 1 accuracy macro avg	0.94 1.00 0.97 0.96	1.00 0.88 0.94 0.96	f1-score 0.97 0.93 0.96 0.95 0.96	74 40 114 114 114
0 1 accuracy macro avg weighted avg	0.94 1.00 0.97 0.96 precision 0.94	1.00 0.88 0.94 0.96 recall	f1-score 0.97 0.93 0.96 0.95 0.96 f1-score 0.97	support  74 40  114 114 114 support  74
0 1 accuracy macro avg weighted avg	0.94 1.00 0.97 0.96 precision	1.00 0.88 0.94 0.96 recall	f1-score 0.97 0.93 0.96 0.95 0.96 f1-score	74 40 114 114 114 support
0 1 accuracy macro avg weighted avg	0.94 1.00 0.97 0.96 precision 0.94	1.00 0.88 0.94 0.96 recall	f1-score 0.97 0.93 0.96 0.95 0.96 f1-score 0.97 0.93	support  74 40  114 114 114 support  74 40
0 1 accuracy macro avg weighted avg	0.94 1.00 0.97 0.96 precision 0.94	1.00 0.88 0.94 0.96 recall	f1-score 0.97 0.93 0.96 0.95 0.96 f1-score 0.97	support  74 40  114 114 114 support  74
0 1 accuracy macro avg weighted avg  0 1	0.94 1.00 0.97 0.96 precision 0.94	1.00 0.88 0.94 0.96 recall	f1-score 0.97 0.93 0.96 0.95 0.96 f1-score 0.97 0.93	support  74 40  114 114 114 support  74 40

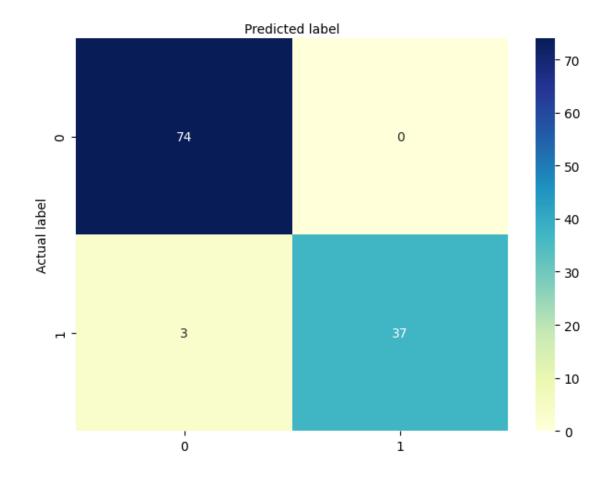
weighted avg	0.96	0.96	0.96	114
	precision	recall	f1-score	support
0	0.94	1.00	0.97	74
1	1.00	0.88	0.93	40
accuracy			0.96	114
macro avg	0.97	0.94	0.95	114
weighted avg	0.96	0.96	0.96	114
	precision	recall	f1-score	support
0	0.95	1.00	0.97	74
1	1.00	0.90	0.95	40
accuracy			0.96	114
macro avg	0.97	0.95	0.96	114
	0.51	0.00	0.00	

#### 22: 0.9736842105263158

	precision	recall	f1-score	support
0	0.96	1.00	0.98	74
1	1.00	0.93	0.96	40
accuracy			0.97	114
macro avg	0.98	0.96	0.97	114
weighted avg	0.97	0.97	0.97	114

```
[]: cnf_matrix = metrics.confusion_matrix(Y_test, Y_pred)
    class_names=[0,1] # name of classes
    fig, ax = plt.subplots()
    tick_marks = np.arange(len(class_names))
    plt.xticks(tick_marks, class_names)
    plt.yticks(tick_marks, class_names)
    # create heatmap
    sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu" ,fmt='g')
    ax.xaxis.set_label_position("top")
    plt.tight_layout()
    plt.title('Confusion matrix', y=1.1)
    plt.ylabel('Actual label')
    plt.xlabel('Predicted label')
```

## Confusion matrix



Compared to the previous questions, based on the f1 scores, this model is slightly worse than 2a, being above 3 and 2b in performance.

## 0.7 Problem 5

```
[]: df = pd.read_csv('cancer.csv')
     df['diagnosis'] = df['diagnosis'].apply(lambda x: 1 if x == 'M' else 0)
     Y = df.iloc[:, 1].values
     df = df.drop(columns='diagnosis', axis=1)
     X = df.iloc[:, 1:].values
[ ]: k = 30
     \max \ accuracy = 0
     max_acc_index = 0
     for n in range(1, k+1):
         pc_list = [f'pc{i}' for i in range(n)]
         pca = PCA(n_components=n)
         principalComponents = pca.fit_transform(X)
         principalDf = pd.DataFrame(data = principalComponents
                     , columns = pc_list)
         X_train, X_test, Y_train, Y_test = train_test_split(principalDf, Y,_
      stest_size = 0.2, random_state = 7)
         sc X = StandardScaler()
         X_train = sc_X.fit_transform(X_train)
         X test = sc X.transform(X test)
         model = GaussianNB()
         model.fit(X_train, Y_train)
         predicted = model.predict(X_test)
         cnf_matrix = confusion_matrix(Y_test, predicted)
         acc = metrics.accuracy_score(Y_test, predicted)
         if acc > max_accuracy:
             max_accuracy = acc
             max_acc_index = n
         print(metrics.classification_report(Y_test, predicted))
     print(f'{max_acc_index}: {max_accuracy}')
```

	precision	recall	il-score	support
0	0.89	0.99	0.94	74
1	0.97	0.78	0.86	40
accuracy			0.91	114
macro avg	0.93	0.88	0.90	114

weighted avg	0.92	0.91	0.91	114
	precision	recall	f1-score	support
0	0.07	0 00	0.00	71
0	0.87	0.99	0.92	74
1	0.97	0.72	0.83	40
accuracy			0.89	114
macro avg	0.92	0.86	0.88	114
weighted avg	0.90	0.89	0.89	114
0 0				
	precision	recall	f1-score	support
0	0.87	0.99	0.92	74
1	0.97	0.72	0.83	40
accuracy			0.89	114
macro avg	0.92	0.86	0.88	114
weighted avg	0.90	0.89	0.89	114
	precision	recall	f1-score	support
0	0.88	0.97	0.92	74
1	0.94	0.75	0.83	40
_	0.02			
accuracy			0.89	114
macro avg	0.91	0.86	0.88	114
weighted avg	0.90	0.89	0.89	114
8				
	precision	recall	f1-score	support
0	0.89	0.99	0.94	74
1	0.97	0.78	0.86	40
_	0.07	0.70	0.00	10
accuracy			0.91	114
macro avg	0.93	0.88	0.90	114
weighted avg		0.91	0.91	114
	precision	recall	f1-score	support
0	0.89	0.97	0.93	74
1	0.94	0.78	0.85	40
1	0.04	0.70	0.00	-10
accuracy			0.90	114
macro avg	0.91	0.87		
weighted avg	0.91		0.90	
3 7 76				<b>-</b>
	precision	recall	f1-score	support

0	0.00	0.07	0.00	7.4
0 1	0.89 0.94	0.97	0.93	74
1	0.94	0.78	0.85	40
accuracy			0.90	114
macro avg	0.91	0.87	0.89	114
weighted avg	0.91	0.90	0.90	114
weighted avg	0.91	0.90	0.90	114
	precision	recall	f1-score	support
0	0.86	0.93	0.90	74
1	0.85	0.72	0.78	40
-	0.00	0.12	0.10	10
accuracy			0.86	114
macro avg	0.86	0.83	0.84	114
weighted avg	0.86	0.86	0.86	114
0	0.00	0.00	0.00	
	precision	recall	f1-score	support
0	0.86	0.92	0.89	74
1	0.83	0.72	0.77	40
1	0.00	0.12	0.11	40
accuracy			0.85	114
macro avg	0.84	0.82	0.83	114
weighted avg	0.85	0.85	0.85	114
#018H004 478	0.00	0.00	0.00	
	precision	recall	f1-score	support
_				
0	0.85	0.91	0.88	74
1	0.80	0.70	0.75	40
			0.00	114
accuracy	0.00	0.00	0.83	
macro avg	0.82	0.80	0.81	114
weighted avg	0.83	0.83	0.83	114
	precision	recall	f1-score	support
0	0.87	0.91	0.89	74
1	0.81	0.75	0.78	40
-	0.01	0.70	0.10	10
accuracy			0.85	114
macro avg	0.84	0.83	0.83	114
weighted avg	0.85	0.85	0.85	114
	precision	recall	f1-score	support
0	0.86	0.89	0.87	74
1	0.78	0.72	0.75	40
1	0.10	0.12	0.75	40

accuracy			0.83	114	
macro avg	0.82	0.81	0.81	114	
weighted avg	•		0.83	114	
weighted avg	0.63	0.83	0.63	114	
	precision	recall	f1-score	support	
	•				
0	0.86	0.91	0.00	74	
0			0.88		
1	0.81	0.72	0.76	40	
accuracy			0.84	114	
•	2 22				
macro avg	0.83	3 0.82 C		114	
weighted avg	0.84	0.84	0.84	114	
0 0					
		7.7	£1		
	precision	recall	f1-score	support	
0	0.89	0.91	0.90	74	
1	0.82	0.80	0.81	40	
accuracy			0.87	114	
•	0.86	0.85	0.85	114	
macro avg					
weighted avg	0.87	0.87	0.87	114	
	precision	recall	f1-score	gunnort	
	precision	recarr	II SCOLE	support	
0	0.87	0.91	0.89	74	
1	0.81	0.75	0.78	40	
_	0.01	0.75	0.70	40	
accuracy			0.85	114	
macro avg	0.84	0.83 0.83		114	
0					
weighted avg	0.85	0.85	0.85	114	
	precision	recall	f1-score	support	
	r				
_					
0	0.86	0.91	0.88	74	
1	0.81	0.72	0.76	40	
			0.04	111	
accuracy			0.84	114	
macro avg	0.83	0.82	0.82	114	
weighted avg	0.84	0.84	0.84	114	
0	0.01	0.02	0.02		
	precision	recall	f1-score	support	
0	0.86	0.89	0.87	74	
1	0.78	0.72	0.75	40	
accuracy			0.83	114	
•	0.00	0.01			
macro avg	0.82	0.81	0.81	114	

weighted avg	0.83	0.83	0.83	114
	precision		f1-score	support
0	0.86	0.89	0.87	74
1	0.78	0.72	0.75	40
1	0.70	0.12	0.75	40
accuracy			0.83	114
macro avg	0.82	0.81 0.81		114
weighted avg		0.83	0.83	114
weighted avg	0.03	0.83 0.83 0.83		114
	precision	recall	f1-score	support
0	0.85	0.91	0.88	74
1	0.80	0.70	0.75	40
accuracy			0.83	114
macro avg	0.82	0.80	0.81	114
weighted avg	0.83	0.83	0.83	114
	precision	recall	f1-score	support
0	0.86	0.89	0.87	74
1	0.78	0.72	0.75	40
accuracy			0.83	114
macro avg	0.82	0.81	0.81	114
weighted avg	0.83	0.83	0.83	114
6				
	precision	recall	f1-score	support
0	0.86	0.89	0.87	74
1	0.78	0.72	0.75	40
accuracy			0.83	114
macro avg	0.82	0.81	0.81	114
weighted avg	0.83	0.83	0.83	114
0 0				
	precision	recall	f1-score	support
0	0.87	0.89	0.88	74
1	0.79	0.75	0.77	40
_				
accuracy			0.84	114
macro avg	0.83	0.82	0.82	114
weighted avg	0.84	0.84	0.84	114
5 44 4 6	<del>-</del>	· · ·		_
	precision	recall	f1-score	support

0	0.87	0.91	0.89	74
1	0.81	0.75	0.78	40
accuracy			0.85	114
macro avg	0.84	0.83	0.83	114
weighted avg	0.85	0.85	0.85	114
	precision	recall	f1-score	support
		2 24		
0	0.87	0.91	0.89	74
1	0.81	0.75	0.78	40
accuracy			0.85	114
macro avg	0.84	0.83	0.83	114
weighted avg	0.85	0.85	0.85	114
weighted avg	0.83	0.00	0.00	114
	precision	recall	f1-score	support
0	0.07	0.01	0.00	7.4
0	0.87	0.91	0.89	74
1	0.81	0.75	0.78	40
accuracy			0.85	114
macro avg	0.84	0.83	0.83	114
weighted avg	0.85	0.85	0.85	114
8			0.00	
	precision	recall	f1-score	support
0	0.87	0.89	0.88	74
1	0.79	0.75	0.77	40
accuracy			0.84	114
macro avg	0.83	0.82	0.82	114
_		0.84	0.84	114
weighted avg	0.84	0.04	0.04	114
	precision	recall	f1-score	support
0	0.88	0.91	0.89	74
1	0.82	0.78	0.79	40
accuracy			0.86	114
macro avg	0.85	0.84	0.84	114
weighted avg	0.86	0.86	0.86	114
	precision	recall	f1-score	support
	-			
0	0.88	0.91	0.89	74
1	0.82	0.78	0.79	40

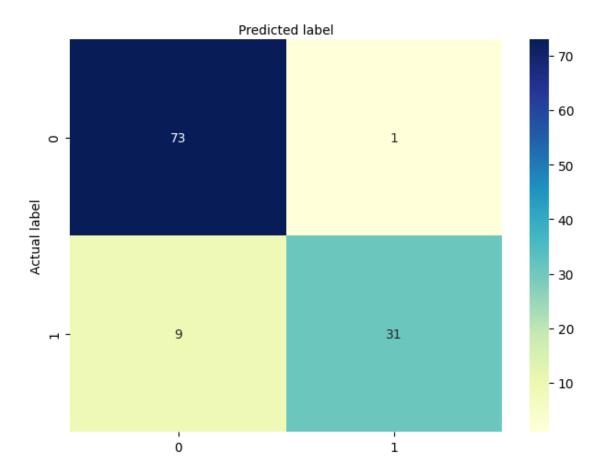
accuracy			0.86	114
macro avg	0.85	0.84	0.84	114
weighted avg	0.86	0.86	0.86	114
	precision	recall	f1-score	support
0	0.88	0.89	0.89	74
1	0.79	0.78	0.78	40
accuracy			0.85	114
macro avg	0.84	0.83	0.84	114
weighted avg	0.85	0.85	0.85	114
	precision	recall	f1-score	support
0	0.88	0.91	0.89	74
1	0.82	0.78	0.79	40
accuracy			0.86	114
macro avg	0.85	0.84	0.84	114
weighted avg	0.86	0.86	0.86	114

## 1: 0.9122807017543859

pr	recision	recall	f1-score	support
0	0.89	0.99	0.94	74
1	0.97	0.78	0.86	40

```
accuracy 0.91 114
macro avg 0.93 0.88 0.90 114
weighted avg 0.92 0.91 0.91 114
```

```
[]: cnf_matrix = metrics.confusion_matrix(Y_test, predicted)
    class_names=[0,1] # name of classes
    fig, ax = plt.subplots()
    tick_marks = np.arange(len(class_names))
    plt.xticks(tick_marks, class_names)
    plt.yticks(tick_marks, class_names)
    # create heatmap
    sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu" ,fmt='g')
    ax.xaxis.set_label_position("top")
    plt.tight_layout()
    plt.title('Confusion matrix', y=1.1)
    plt.ylabel('Actual label')
    plt.xlabel('Predicted label')
```



This model had the worst f1 scores, being slightly below 2b.